TYPESET-10

COMPOSITION COMMANDS MANUAL

TYPESET-10
COMPOSITION COMMANDS

SERIES 200
PRELIMINARY
September 1, 1974

This manual is supplied for information only and is subject to change without notice. Please direct comments concerning this manual to:

Digital Equipment Corporation
Typeset-10 Product Line
Marlboro, Massachusetts

© COPYRIGHT 1974, DIGITAL EQUIPMENT CORPORATION
MARLBORO, MASSACHUSETTS

The information in this document is subject to change without notice and should not be construed to be a commitment by Digital Equipment Corporation. Actual distribution of the software to provide the capability described in this document will be subject to Terms and Conditions specified in the DECsystem-10/Typeset-10 Software License for such software. Digital Equipment Corporation assumes no responsibility for the use or reliability of its software on equipment which is not supplied by Digital Equipment Corporation. This software and documentation is furnished to purchaser under a license to use on a single computer system and can be copied (with inclusion of DEC's copyright notice) only for use in such system, except as may otherwise be provided in writing by DEC.

This document was composed using the TYPESET 10 SYSTEM and typeset on a Graphic System, Inc. C/A/T photocomposer at Digital Equipment Corporation, Marlboro, Massachusetts.

CONTENTS

			PAGE
CHAPTER	**1**	**INTRODUCTION**	1-1
	1.1	FUNCTION AND USE OF COMPOSITION COMMANDS	1-1
	1.2	HOW COMPOSITION COMMANDS ARE PROCESSED	1-1
	1.3	SYSTEM STANDARDS	1-2
	1.4	COMPOSITION COMMAND FORMATS	1-3
	1.5	CONCATENATION	1-4
	1.6	INDENT NOTATION - # CONVENTION	1-4
	1.7	POINT SIZE/SET SIZE NOTATION	1-5
	1.8	LEADING NOTATION	1-5
	1.9	COLUMN WIDTH (MEASURE) NOTATION	1-5
	1.10	FORMAT OF THE MANUAL AND NOMENCLATURE	1-5
CHAPTER	**2**	**ESSENTIAL PARAMETERS**	2-1
	2.1	SETUP	2-1
	2.2	TYPE FACE (FONT)	2-1
	2.3	POINT SIZE	2-2
	2.4	STORED LEADING	2-2
	2.5	COMBINATION POINT SIZE AND LEADING COMMAND	2-4
	2.6	COLUMN WIDTH (MEASURE)	2-4
	2.7	TYPICAL WAYS OF SPECIFYING ESSENTIAL PARAMETERS	2-4
CHAPTER	**3**	**TYPEFACE CHANGE COMMAND**	3-1
	3.1	CHANGE FACE	3-1
	3.2	BOLD FACE	3-1
	3.3	ITALIC FACE	3-2
	3.4	SMALL CAPS FACE	3-3
	3.5	FACE RESTORE	3-4
	3.6	FACE MODIFY (SLANT)	3-4
	3.7	CANCEL SLANT	3-4
	3.8	CHANGE FACE ONE-LINE	3-4
	3.9	BOLD FACE ONE-LINE	3-5
	3.10	ITALIC ONE-LINE	3-5
	3.11	SMALL CAPS ONE-LINE	3-6
	3.12	ALL CAPS ONE-LINE	3-6
CHAPTER	**4**	**POINT SIZE AND SET SIZE CHANGE COMMANDS**	4-1
	4.1	CHANGE POINT SIZE	4-1
	4.2	CHANGE POINT SIZE ONE-LINE	4-1
	4.3	CHANGE SET SIZE	4-2
	4.4	CHANGE SET SIZE ONE-LINE	4-3
	4.5	RESTORE SET SIZE	4-3

PAGE

CHAPTER 5 LEADING CHANGE COMMANDS 5-1
5.1 CHANGE LEADING 5-1
5.2 CHANGE LEADING ONE-LINE 5-1
5.3 CHANGE VERTICAL DIVISION (PROPORTIONAL LEADING) . 5-2
5.4 ADDED LEAD 5-3
5.5 REVERSE LEAD 5-3
5.6 CANCEL LEADING ONE-LINE 5-4

CHAPTER 6 COLUMN WIDTH CHANGE COMMANDS 6-1
6.1 CHANGE COLUMN WIDTH 6-1
6.2 COLUMN RESTORE 6-1

CHAPTER 7 QUAD COMMANDS 7-1
7.1 QUAD LEFT 7-1
7.2 QUAD RIGHT 7-1
7.3 QUAD CENTER 7-1
7.4 QUAD MIDDLE (SPLIT QUAD) 7-2
7.5 CONDITIONAL QUAD 7-2

CHAPTER 8 PARAGRAPH FLUSH AND HANG COMMANDS 8-1
8.1 CHANGE PARAGRAPH INDENT 8-1
8.2 PARAGRAPH INDENT 8-1
8.3 RESTORE PARAGRAPH INDENT 8-1
8.4 DEFINE FLUSH AND HANG INDENT 8-2
8.5 FLUSH AND HANG 8-2
8.6 CANCEL FLUSH AND HANG 8-3

CHAPTER 9 SPACING COMMANDS 9-1
9.1 EM SPACE 9-1
9.2 EN SPACE 9-1
9.3 THIN SPACE 9-1
9.4 FIXED SPACE 9-1
9.5 FIGURE SPACE 9-2
9.6 ONE-UNIT SPACE 9-2
9.7 THUT (THIN-PLUS) 9-2
9.8 BOLT (EN-PLUS) 9-2
9.9 EM DASH 9-2

CHAPTER 10 PI CASE COMMANDS 10-1
10.1 SINGLE PI CHARACTER 10-1
10.2 ENTER PI CASE 10-1
10.3 EXIT PI CASE 10-2

CHAPTER 11 TAB COMMANDS 11-1
11.1 SET TAB STOPS 11-1
11.2 SET EQUAL TAB STOPS 11-2
11.3 SET PROPORTIONAL TAB STOPS 11-2
11.4 SET TAB IN TEXT 11-3
11.5 TAB LEFT 11-3

PAGE

	11.6	TAB RIGHT	11-4
	11.7	TAB CENTER	11-4
	11.8	TAB AROUND	11-5
	11.9	NOTES AND CAUTIONS FOR USING TABULATION COMMANDS	11-5
	11.10	CANCEL TAB STOPS	11-6
CHAPTER	12	INDENT COMMANDS	12-1
	12.1	COLUMN INDENT COMMANDS	12-3
	12.1.1	Indent Left	12-3
	12.1.2	Indent Right	12-4
	12.1.3	Indent Both	12-4
	12.1.4	Fixed Indent	12-5
	12.1.5	Cancel Left Indent	12-6
	12.1.6	Cancel Right Indent	12-7
	12.1.7	Cancel Both Indents	12-7
	12.2	TEXT INDENT COMMANDS	12-7
	12.2.1	Define Text Indent	12-7
	12.2.2	Left Text Indent	12-8
	12.2.3	Right Text Indent	12-9
	12.2.4	Cancel Left Text Indent	12-10
	12.2.5	Cancel Right Text Indent	12-10
	12.3	HANGING INDENT COMMANDS	12-10
	12.3.1	Left Hanging Indent	12-10
	12.3.2	Right Hanging Indent	12-12
	12.3.3	Hanging Indent Both	12-13
	12.3.4	Fixed Hanging Indent	12-14
	12.3.5	Cancel Left Hanging Indent	12-14
	12.3.6	Cancel Right Hanging Indent	12-14
	12.3.7	Cancel Both Hanging Indents	12-15
	12.4	LENGTH FIT COMMANDS	12-15
	12.4.1	Left Length Fit	12-15
	12.4.2	Right Length Fit	12-16
	12.4.3	Length Fit Both	12-17
	12.4.4	Cancel Left Length Fit	12-19
	12.4.5	Cancel Right Length Fit	12-19
	12.4.6	Cancel Length Fit Both	12-20
	12.5	SKEW COMMANDS	12-21
	12.5.1	Forward Left Skew	12-21
	12.5.2	Reverse Left Skew	12-22
	12.5.3	Forward Right Skew	12-23
	12.5.4	Reverse Right Skew	12-24
	12.5.5	Forward Skew Both	12-25
	12.5.6	Reverse Skew Both	12-27
	12.5.7	Cancel Left Skew	12-27
	12.5.8	Cancel Right Skew	12-28
	12.5.9	Cancel Skew Both	12-28
	12.5.10	Cancel All Indents Command	12-28

PAGE

CHAPTER 13 MARK POINT COMMANDS 13-1
13.1 SET VERTICAL MARK 13-1
13.2 CANCEL VERTICAL MARK 13-1
13.3 CANCEL ALL MARKS 13-2
13.4 VERTICAL POSITION 13-2
13.5 VERTICAL COLUMNS 13-2
13.6 CANCEL VERTICAL COLUMNS 13-4
13.7 VERTICAL BOTH 13-4
13.8 VERTICAL GUTTER 13-5

CHAPTER 14 FRACTION COMMANDS 14-1
14.1 ONE-EIGHTH (1/8) 14-1
14.2 ONE-FOURTH (1/4) 14-1
14.3 THREE-EIGHTHS (3/8) 14-1
14.4 ONE-HALF (1/2) 14-2
14.5 FIVE-EIGHTHS (5/8) 14-2
14.6 THREE-FOURTHS (3/4) 14-2
14.7 SEVEN-EIGHTHS (7/8) 14-2
14.8 ONE-THIRD (1/3) 14-3
14.9 TWO-THIRDS (2/3) 14-3

CHAPTER 15 RAGGED MODE COMMANDS 15-1
15.1 RAGGED LEFT 15-1
15.2 RAGGED RIGHT 15-2
15.3 RAGGED CENTER 15-2
15.4 RAGGED MIDDLE 15-3
15.5 CANCEL RAGGED MODE 15-4

CHAPTER 16 OUTPUT COMMANDS 16-1
16.1 ENABLE AUTO RULE DROP 16-1
16.2 CANCEL AUTO RULE DROP 16-1
16.3 CANCEL FLASH 16-2
16.4 RESTORE FLASH 16-2
16.5 CANCEL FLASH AND LEAD (ONE-LINE) 16-2
16.6 SELECT OUTPUT DEVICE 16-3
16.7 SET DELAY 16-3
16.8 RESET DELAY 16-4
16.9 OUTPUT DELAY 16-4

CHAPTER 17 LEADER COMMANDS 17-1
17.1 USE LEADERS 17-1
17.2 DEFINE OTHER LEADERS 17-1
17.3 USE OTHER LEADERS 17-2
17.4 CANCEL LEADER MODE 17-2
17.5 ONE-SHOT LEADERS 17-3
17.6 ONE-SHOT OTHER LEADERS 17-3
17.7 ONE-SHOT RULE 17-4
17.8 ONE-SHOT WHITE SPACE 17-4

PAGE

CHAPTER	**18**	**JUSTIFICATION COMMANDS**	18-1
	18.1	JUSTIFICATION PARAMETERS	18-1
	18.2	ALLOW LETTER SPACING	18-2
	18.3	CANCEL LETTER SPACING	18-3
	18.4	ONE-LINE JUSTIFY	18-4
CHAPTER	**19**	**HYPHENATION COMMANDS**	19-1
	19.1	HYPHENATION PARAMETERS	19-1
	19.2	DISABLE HYPHENATION	19-2
	19.3	ENABLE HYPHENATION	19-4
	19.4	ONE-WORD DISABLE	19-4
	19.5	USE DICTIONARY ONLY	19-6
	19.6	DISCRETIONARY HYPHEN	19-6
CHAPTER	**20**	**MESSAGE COMMANDS**	20-1
	20.1	TAG LINE MESSAGE	20-1
	20.2	PUNCH MESSAGE	20-1
	20.3	BOTH MESSAGES	20-2
CHAPTER	**21**	**MACROS**	21-1
	21.1	WHAT IS A MACRO?	21-1
	21.2	WHEN TO USE MACROS	21-1
	21.3	HOW TO DEFINE A MACRO	21-1
	21.4	HOW TO CALL A MACRO	21-2
	21.5	MACROS WITH SEVERAL PARTS: MERGING TEXT	21-3
	21.6	REPEATING A MACRO CALL	21-5
	21.7	NESTED MACROS	21-6
CHAPTER	**22**	**ARITHMETIC COMMANDS**	22-1
	22.1	COMMAND SYNTAX	22-1
	22.2	LOAD REGISTER	22-1
	22.3	DISPLAY REGISTER (ARABIC)	22-2
	22.4	DISPLAY REGISTER (WORDS)	22-2
	22.5	ZERO REGISTER	22-2
	22.6	INCREMENT REGISTER	22-2
	22.7	DISPLAY AND INCREMENT REGISTER	22-3
	22.8	ADD TO REGISTER	22-3
	22.9	MULTIPLY REGISTER	22-3
	22.10	DIVIDE REGISTER	22-4
	22.11	REMAINDER	22-4
	22.12	USING ARITHMETIC COMMANDS	22-4
APPENDIX	**A**	**ERROR MESSAGES**	A-1
APPENDIX	**B**	**USING THE HYPHENATION EXCEPTION WORD DICTIONARY**	B-1
SUMMARY INDEX BY FUNCTION			Index-1

FIGURES

Page

Figure 12-1 User Address Mapping 12-2

CHAPTER 1

INTRODUCTION

1.1 FUNCTION AND USE OF COMPOSITION COMMANDS

Typeset-10 composition commands control the appearance of typeset copy and select the setup for allotting copy to a typesetting machine. Composition commands are included within the text of copy, which may be input to Typeset-10 via the editing programs COPYED and PRUFED or the wire service and local keyboard translator WIPTIN.

Composition commands may specify:

1. Setup.
2. Point size and set size.
3. Typeface: including bold face, italic, small caps, and slant.
4. Leading: including stored leading, reverse lead, vertical mark points, and multiple column setting.
5. Column width.
6. Quads and Tabs.
7. Pi case characters, fixed space, and fractions.
8. Indents: including paragraph indents, hanging indents, length fits and skews.
9. Justification and hyphenation parameters.
10. Output messages for film and paper tape.

The composition commands also include a facility for defining and using macros. A macro definition is a sequence of composition commands or text which is marked up in a particular format and inserted in copy. The macro is used by calling its name within the copy. The contents of the macro definition are inserted into the typeset copy at the point of the call. Once a macro is defined, it is stored in the Typeset-10 system and may be called in subsequent takes.

1.2 HOW COMPOSITION COMMANDS ARE PROCESSED

Composition commands are processed by the justification and hyphenation (J & H) program, called JUSTIF. In addition to producing justified and hyphenated lines of copy for typesetting, JUSTIF performs the formatting functions specified by the composition commands. JUSTIF runs continually in the Typeset-10 system. The input to JUSTIF is copy (usually containing composition commands) stored on disk as a DECsystem-10 file. Each file is identified by its filename and

extension (see DECsystem-10 Users Handbook). The filename is created by COPYED, PRUFED, or WIPTIN. The extensions are .CED, for COPYED files, .PED for PRUFED files, and .WIR or .KBD for WIPTIN files. These three programs, either automatically or on command, submit the copy files to JUSTIF for processing. (This submittal is called "queuing" in the DECsystem-10.) JUSTIF processes the composition commands, justifies and hyphenates the copy, and may produce two output files:

1. A file with the original filename and extension .JH. The .JH file contains the text (ASCII) of the original copy after justification, hyphenation, and formatting according to the composition commands. The .JH file also contains any error messages pertaining to improper use of composition commands, and a summary of the number of lines, keystrokes, words and leading. The .JH file is used by COPYED and PRUFED when the user requests a line-for-line proof on the line printer.

2. A file with the original filename and extension .BIN. The .BIN file contains the actual characters for typesetting in the appropriate code for the typesetting machine in the specified setup. A .BIN file is actually generated by a program called an output driver which is used by JUSTIF. JUSTIF uses different output drivers, which are designed for specific typesetting machines, depending on the machine implied by the setup command. The .BIN file is used by ALLOT to set type.

1.3 SYSTEM STANDARDS

System standards for JUSTIF are established when the program is assembled (see TSYSCF.MAC File Specification). System standards include:

1. Default justification parameters and letter spacing (Chapter 18).

2. Default hyphenation parameters (Chapter 19).

3. Default essential parameters (Chapter 2).

4. Default paragraph indent (Chapter 8).

5. Default ragged middle line (Chapter 15).

6. Fixed indents and hanging indents (Chapter 12).

7. Inclusion of mark point commands (Chapter 13).

8. Type of stored leading usage (Chapter 2).

9. Parameters for setting slug line.

10. Maximum number of grids, type faces, and point sizes handled by JUSTIF.

11. Type of rule drop handling (Chapter 16).

12. Indent notation usage for # symbol (explained later in this chapter).

The first five items (default parameters) may be overridden within a take by composition commands, as explained in the manual where applicable. The remaining system standards are fixed.

1.4 COMPOSITION COMMAND FORMATS

There are two basic command formats: the first uses the slash character and the second uses the square bracket characters.

Slash commands consist of a slash followed by a single letter or number. These commands have functions similar to the single-key functions of certain keyboards; examples are quad left (/l), em space (/m), the fraction 1/8 (/1).

Square bracket commands consist of a left square bracket, the command itself, and a right square bracket. If commands are concatenated (see below), the terminating right bracket may be omitted from all but the last command. The command between the brackets consists of one or two letters and may also contain numeric information, often several numbers separated by commas. Macro commands are special cases of square bracket commands; their syntax is described in Chapter 21.

NOTE

If a terminating square bracket is omitted by mistake, JUSTIF terminates the command at the first non-numeric character and interprets the command. Sometimes the interpretation is correct, for example [p8Today will set "Today" in 8-point type. If a non-numeric character is typed by mistake within a command, JUSTIF also terminates and interprets the command. The interpretation usually produces incorrect typesetting. For example, if the letter t is typed instead of the number 5 in the command [hl3,t,20], type will be set with a left hanging indent of 3 picas beginning immediately and lasting forever. The beginning of the hanging "text" would be interpreted to be "t,20]". In either case, JUSTIF issues no error message if any legal interpretation can be made.

Because the slash and square bracket characters are used in composition command formats, these characters cannot be directly accessed for text. For inclusion in text, slash and square bracket codes must be defined as special characters in the font tables or accessed from Pi case (see FONGEN Manual).

Composition commands may be typed in either upper-case or lower-case. Most commands have a mnemonic relationship to their function; for example, f for font commands, i for indent commands, l for length fit commands, p for pointsize commands, and v (vertical) for leading commands. The symbol % in commands usually means cancel a previous command or restore a default value.

1.5 CONCATENATION

To minimize keystrokes when several commands are issued in a row, square bracket commands may be concatenated, i.e., strung together. To concatenate commands, omit all but the final right square bracket. An example of concatenated commands is [p10[c15].

1.6 INDENT NOTATION - # CONVENTION

In certain composition commands, indents (horizontal offsets from the left/right column edge) can be specified in either a standard or an alternate notation. The alternate notation is preceded by the symbol #.

There are three possible conventions for standard and alternate indent notation. A convention is established in the system standards and is permanent for a given installation of JUSTIF. The three conventions are:

1. Standard: picas and points, separated by a period.
   Alternate: ems and ens, separated by a period.
   Examples: [il3] left indent of 3 picas
   [il3.5] left indent of 3 picas, 5 points
   [il#3] left indent of 3 ems
   [il#3.1] left indent of 3 ems and 1 en

   NOTE
   [il#2.3] would be equivalent to a left indent of 3 ems and 1 en.

2. Standard: picas and points as in 1.
   Alternate: ens
   Examples: [il3.5] left indent of 3 picas, 5 points as above.
   [il#7] left indent of 7 ens

3. Standard: ems and ens, separated by a period.
   Alternate: picas and points separated by a period.
   Examples: [il3] left indent of 3 ems
   [il3.5] left indent of 3 ems, 5 ens
   [il#3] left indent of 3 picas
   [il#3.5] left indent of 3 picas, 5 points

1.7 POINT SIZE/SET SIZE NOTATION

Pointsize and setsize are specified in points and decimal fractions of points, up to two decimal places. That is, the decimal point has its usual mathematical meaning. For example, [p8.5] and [p8.50] both specify 8½ point type.

1.8 LEADING NOTATION

Leading is specified in points and decimal fractions of points, the same as point size. Examples:

[v8.05] means 8 5/100 points of stored lead.

Both [v8.5] and [v8.50] mean 8½ points of stored lead.

1.9 COLUMN WIDTH (MEASURE) NOTATION

Column width is always specified in picas or picas and points. If points are included, the picas and points are separated by a period. For example,

[c11]	Specifies an 11-pica measure
[c11.6]	specifies an 11½-pica measure (11 picas, six points)
[c11.5]	specifies an 11-pica, 5-point measure.

In other words, column width notation is always the same as the first example of standard indent notation above.

1.10 FORMAT OF THE MANUAL AND NOMENCLATURE

Each section of this manual describes a functional group of composition commands. The description of each command contains the format and function of the command, when it is effective and terminated, and in most cases an example.

In the formats, information entered literally as shown is printed in standard face, while variable information (usually numeric) is printed in italics; variable information is referred to the appropriate notation defined in this section. For example, the column width command format is shown as

[c*x*]

where *x* is column width notation. Thus, the information following "c" could be 12 (12 picas) or 12.5 (12 picas, 5 points). The pointsize command format is shown as

[p*x*]

where *x* is pointsize notation. Thus, the information following "p" could be 12 (12-point) or 12.5 (12½-point). Under "When Effective", the word "line" always means an output line of typeset

copy. (Unless a line-ending command like a quad immediately precedes some text, it is sometimes difficult to predict the beginning of a line by inspecting the marked-up copy.) The word "immediately" is used often, and always means as soon as the command is processed and found to be legal.

All commands are terminated at the end of a take. Therefore, this is not explicitly stated for every command under "When Terminated."

The examples consist of both a typed sample of marked-up copy (as in a COPYED file) and a specimen of the resulting typeset copy. The "false hyphens" <-> which appear in the examples of marked-up copy, are inserted by the input programs and are not typed by the keyboarder.

Chapter 21 describes the construction and use of macros. The Appendices describe the use of an exception word hyphenation dictionary and the JUSTIF error messages with associated corrective action.

CHAPTER 2

ESSENTIAL PARAMETERS

Five essential parameters are required to typeset copy:

1. ALLOT setup number.
2. Type face (font)
3. Point size
4. Leading
5. Column width

If any essential parameter is not explicitly selected by a composition command at the beginning of a take, the default value in the system standards is used. It is possible to omit all commands for essential parameters and rely only on default values, for example in wire service (WIPTIN) copy. The commands for selecting essential parameters are described below.

2.1 SETUP

Format: [d*x*]

x is a setup number defined via the ALLOT program.

Function: This command selects the ALLOT setup for typeset output. If specified for a take, which is usual, the setup command must precede all other commands and copy. Only one setup command may appear in a take. If the setup number is not defined in ALLOT (See ALLOT Manual), JUSTIF ignores the command, issues an error message, and uses the default setup number.

When Effective: Immediately.

When Terminated: At end of take.

Example: [d1] specifies setup 1.

2.2 TYPE FACE (FONT)

Format: [f*x*]

x is a type face number defined via the FONGEN program.

Function: This command selects the initial typeface. The face may be changed later in the take by another [f command or any of the commands in Chapter 3. If the typeface number is not included in the current

ALLOT setup, JUSTIF ignores the command, issues an error message, and uses the default type face.

When Effective: Immediately.

When Terminated: By another [f command or a type face change command described in Chapter 3:

1. [wf
2. [fb or /e
3. [wb
4. [fi
5. [wi
6. [fs
7. [ws
8. [wc

Example: [f20] specifies type face 20.

2.3 POINT SIZE

Format: [p*x*]

x is the point size, written in point size notation (See Chapter 1). If a fractional point size is impossible on the current typesetting machine, it is rounded down to the next acceptable fraction.

Function: This command selects the initial point size. The point size may be changed later in the take by another [p command. If the point size is not available in the current ALLOT setup, JUSTIF ignores the command, issues an error message, and uses the default point size.

NOTE

The point size command also implicitly selects the set size (width), as described in Chapter 4.

When Effective: Immediately.

When Terminated: By another [p command or [wp command (See Chapter 4).

Example: [p10.5] selects 10½ point type.

2.4 STORED LEADING

Format: [v*x*]

x is the stored leading, expressed in leading notation (See Chapter 1). If a fractional amount of leading is not acceptable for the output

typesetting machine, the leading is rounded down to the next available fraction.

Function: This command selects the initial leading between lines of copy (also called stored leading). The leading may be changed later in the take by another [v command.

JUSTIF uses one of four possible methods of applying the stored leading. The method is selected in the system standards and is permanent for each installation of JUSTIF. The four methods are:

1. Leading after a line is set.

2. Leading before a line is set.

3. Leading before a line is set, using the maximum leading specified on the line. This method is different from number 2 only when the leading is changed.

4. Leading before a line is set, using a percentage of the maximum leading for the previous line plus a percentage of the maximum leading for the current line. This method is different from method 2 only when the leading is changed on two consecutive lines. The leading proportion is established in the system standards, and may be changed by a [vd command (Chapter 5). Example:
   The system standard is set for proportional leading (before) in the ratio 1/3 - 2/3. The leading in effect for a particular line is 6 points; the leading is changed on the next line to 12 points: [v12]. Therefore, the vertical space between these two lines will be 10 points (1/3 X 6 + 2/3 X 12).

   JUSTIF applies the stored leading by the method selected for the system standards, regardless of the physical characteristics of the output typesetting machine.

When Effective: Either before or after the line containing the command, depending on the system standard.

NOTE

Line always refers to the output line in the typeset copy, not the input line in which the command appears.

When Terminated: By another [v command or (for one line only) by a [wv, [vz, or [ob command.

Examples: [v12] specifies 12 point leading.

[v12.5] specifies 12 and ½ point leading.

2.5 COMBINATION POINT SIZE AND LEADING COMMAND

Format: [p*x*,*y*]

x is the point size in the notation above.

y is the leading in the notation above.

Function: Leading may be added to the point size command, using a comma.

Example: [p10,12] specifies "10 on 12."

2.6 COLUMN WIDTH (MEASURE)

Format: [c*x*]

x is the column width, expressed in column width notation (See Chapter 1).

Function: This command selects the initial column width. The column width may be changed later in the take. If the specified column width is not permitted in the current setup, JUSTIF ignores the command, issues an error message, and uses the default column width if the specified measure was too small, or the maximum column width if the specified measure was too large.

When Effective: At the beginning of a line - effective immediately. In the middle of a line - effective at the beginning of the next line.

When Terminated: By another [c command or a [cz command.

Example: [c11.6] sets the column width to 11 picas, 6 points.

2.7 TYPICAL WAYS OF SPECIFYING ESSENTIAL PARAMETERS

Usually, the essential parameters are specified in concatenated form at the beginning of copy for a take. Example:

[d1[f9[p10[v11[c11.6]

Alternately, a macro containing the essential parameters may be defined as explained in Chapter 21. Then just the macro call is entered at the beginning of the copy. For example, a macro may be defined once by

[*S2=[d2[f3[p10,12[c11]=]

Then all subsequent takes using these essential parameters may begin with the macro call [*S2].

CHAPTER 3

TYPEFACE CHANGE COMMANDS

3.1 CHANGE FACE

Format: [f*x*]

x is a type face number.

Function: Same command as essential parameter specification. Changes the previous typeface.

Face *x* remains in effect until changed. If the typeface number is not included in the current ALLOT setup, JUSTIF ignores the command and issues an error message.

When Effective: Immediately

When Terminated: By any type face change command.

1. [f
2. [wf
3. [fb or /e
4. [wb
5. [fi
6. [wi
7. [fs
8. [ws
9. [wc
10. [fm

Example:

```
[c20]<=>
[f405]'48 DeSoto - [f401]6,000 orig.
mi., new eng. and trans.<=>
```

'48 DeSoto - 6.000 orig. mi., new eng. and trans.

3.2 BOLD FACE

Format: [fb] or /e

Function: Sets following text in the bold face of the current face number (*x*), which was selected in the last f command. The bold face must be defined for the current face in FONGEN. If the bold face is not included in the current ALLOT set-up, JUSTIF ignores the command, issues an error message and continues to use the face in effect.

NOTE

Because the bold face is defined in FONGEN with a unique face number (like the light face), the bold face could alternatively be selected with a normal f command. For example, if Times Roman is face 2 and Times Roman Bold is face 20 then Bold face could be selected by an [f20] command instead of [fb]. However, the light face may later be restored only by another [f2] command and not by [fr].

When Effective: Immediately

WhenTerminated:
1. By an [fr] or /d command.
2. By an [f*x*] (change face) command.
3. By [fi]
4. By [fs]

Example:

```
[c20]<->
Co-featured was [FB]The Herbert Hoover
Story,[FR] with music by Curtis Mayfield.<->
```

Co-featured was **The Herbert Hoover Story**, with music by Curtis Mayfield.

3.3 ITALIC FACE

Format: [fi]

Function: Sets following text in italic face of current face number (*x*), selected in previous [f command. The italic face must be defined for the current face in FONGEN. If the italic face is not included in the current ALLOT set-up, JUSTIF ignores the command, issues an error message, and continues to use the face in effect.

NOTE

The italic face could also be selected explicitly by using its face number in the normal [f command. However, the light face may later be restored only by another [f command and not by [fr].

When Effective: Immediately

When Terminated:
1. By an [fr] or /d command.
2. By an [f*x*] (change face) command.
3. By [fb]
4. By [fs]

Example:

```
[c20]<=>
Defense attorneys entered a plea of
[fi]nolo contendere[fr] to the bribery
charge.<=>
```

Defense attorneys entered a plea of *nolo contendere* to the bribery charge.

3.4 SMALL CAPS FACE

Format: [fs]

Function: Sets following text in small caps face of current face number (*x*), as defined in FONGEN. If small caps are not included in the current ALLOT setup, JUSTIF ignores the command, issues an error message, and continues to use the face in effect.

NOTE

The small caps face could also be selected explicitly by using its face number in the normal [f command. However, the light face may later be restored only by another [f command and not by [fr].

When Effective: Immediately

When Terminated:
1. By an [fr] or /d command.
2. By an [f*x*] (change face) command.
3. By [fb]
4. By [fi]

Example:

```
[c20]<=>
The number of [fs]no gas[FR] signs
was exceeded only by the number of
angry motorists.<=>
```

The number of NO GAS signs was exceeded only by the number of angry motorists.

3.5 FACE RESTORE

Format: [fr] or /d

Function: Cancels any [fb, [fi, or [fs in effect. Restores the face number in the previous [f*x*] command.

When Effective: Immediately

Example: See previous two examples.

3.6 FACE MODIFY (SLANT)

Format: [fm]

Function: Sets following text with a slant applied to current face. If the output machine has no slant capability, the command is ignored.

When Effective: Immediately

When Terminated: By [fm%] command.

3.7 CANCEL SLANT

Format: [fm%]

Function: Cancels [fm] command.

When Effective: Immediately

3.8 CHANGE FACE ONE-LINE

Format: [wf*x*]

x is a type face number

Function: Same as [f command, but effective for maximum of one output line. If issued in the middle of a line, the remainder of the line is set in the new face.

When Effective: Immediately

When Terminated: End of line; previous face restored.

Example:

```
[c20]<->
LIFELIKE PLASTIC FLOWERS/c[wt405]<->
MIX 'N' MATCH/c<->
SALE ENDS FRI/c<->
```

LIFELIKE PLASTIC FLOWERS
MIX 'N' MATCH
SALE ENDS FRI

3.9 BOLD FACE ONE-LINE

Format: [wb]

Function: Same as [fb command, but effective for maximum of one output line. If issued in the middle of a line, the remainder of the line is set in bold face.

When Effective: Immediately

When Terminated: End of line; previous face restored.

Example:

```
[c20]<->
All seven occupants were rescued over
ladders by firemen./l<->
[wb]Neighbor Saw Smoke/c<->
```

All seven occupants were rescued over ladders by firemen.

Neighbor Saw Smoke

3.10 ITALIC ONE-LINE

Format: [wi]

Function: Same as [fi command, but effective for maximum of one output line. If issued in the middle of a line the remainder of the line is set in italic.

When Effective: Immediately

When Terminated: End of line; previous face restored.

Example:

```
Westchester Wingnuts, Inc./c[wl]<->
An Equal Opportunity Employer/c<->
Shangri-la Industrial Park/c<->
```

Westchester Wingnuts, Inc.
An Equal Opportunity Employer
Shangri-la Industrial Park

3.11 SMALL CAPS ONE-LINE

Format: [ws]

Function: Same as [fs command, but effective for maximum of one output line. If issued in the middle of a line, the remainder of the line is set in small caps.

When Effective: Immediately

When Terminated: End of line; previous face restored.

Example:

```
[c20]<->
All seven occupants were rescued over
ladders by firemen./l[ws]<->
Neighbor saw smoke/c<->
```

All seven occupants were rescued over ladders by firemen.
NEIGHBOR SAW SMOKE.

3.12 ALL CAPS ONE-LINE

Format: [wc]

Function: Sets one line or portion of line in upper-case. If issued in the middle of a line, the remainder of the line is set in caps.

When Effective: Immediately

When Terminated: End of line.

Example:

```
House for Sa|e/c[wc]s|x-room cape/c
M|d 30's/c<->
```

House for Sale
SIX–ROOM CAPE
Mid 30's

CHAPTER 4

POINT SIZE AND SET SIZE CHANGE COMMANDS

4.1 CHANGE POINT SIZE

Format: [p*x*]

x is the point size, expressed in point size notation (see Chapter 1).

Function: Same command as essential parameter. If the point size is not available, JUSTIF issues an error message and ignores the command. Point size remains in effect until changed.

NOTE
The Change Point Size command also implicitly selects the set size (width), as described in Chapter 4,

When Effective: Immediately

When Terminated: By another [p or [wp command.

Example:

```
[p12]Many[p18] BIG[p12] Values/c<->
```

Many BIG Values

4.2 CHANGE POINT SIZE ONE-LINE

Format: [wp*x*]

x is the point size, expressed in point size notation (see Chapter 1).

Function: Same as [p (Change Point Size) command, but effective for maximum of one output line. If issued in the middle of a line, the remainder of the line is set in the new point size.

When Effective: Immediately

When Terminated: End of line; previous point size restored.

Example:

```
[p18]BEER STRIKE AVERTED/c<->
[wp6]Special to the Maynard Mirror/c<->
```

BEER STRIKE AVERTED

Special to the Maynard Mirror

4.3 CHANGE SET SIZE

Format: [es*x*]

x is the set size (character width) in set size notation (see Chapter 1).

Function: This command changes the set size for following text. The set size is also implicitly specified by the [p point size command. For example [p12] selects 12 point type and 12 point set width. A point size change following an [es command overrides the set size change.

If the output machine cannot change set size, JUSTIF ignores the command, issues an error message, and uses the current set size.

When Effective: Immediately

When Terminated:
1. By [es%] command.
2. By a [p command.

Example:

```
[C20[P18,20[FS15]SALE! SALE! SALE!/C<->
[P14]EVERYTHING MUST GO/C<->
```

SALE! SALE! SALE!
EVERYTHING MUST GO

4.4 CHANGE SET SIZE ONE-LINE

Format: [wex]

x is the set size in set size notation (see Chapter 1).

Same as [es command, but effective for maximum of one output line. If issued in the middle of a line, the remainder of the line is set using the new set size; however, if the output machine cannot change set size in the middle of a line, the command is ignored.

When Effective: Immediately

When Terminated: End of line, previous set size restored.

Example:

```
[p1a,20[we151SALE! SALE! SALE!/C<->
EVERYTHING MUST GO/C<->
```

SALE! SALE! SALE!
EVERYTHING MUST GO

4.5 RESTORE SET SIZE

Format: [es%]

Function: Cancels effect of current [es command. Restores set size implied by last [p command.

When Effective: Immediately

Example:

```
[p1a,20[es241SALE! SALE! SALE![es%1/C<->
EVERYTHING MUST GO/C/L/L/L<->
```

SALE! SALE! SALE!
EVERYTHING MUST GO

CHAPTER 5

LEADING CHANGE COMMANDS

5.1 CHANGE LEADING

Format: [v*x*]

x is the stored leading, expressed in leading notation (See Chapter 1). If a fractional amount of leading is not acceptable for the typesetting machine, the leading is rounded down to the next available fraction.

Function: Same command as essential parameter specification. Stored leading remains in effect until changed.

When Effective: Either before or after the line containing the command, depending on the system standard.

When Terminated: By another [v command or (for one line only) by [wv, [vz, or [ob.

Example:

```
[v20][wb]PRODUCE/c<=>
[v14]LETTUCE $.29 hd./l<=>
TOMATOES $.39 lb./l<=>
```

PRODUCE

LETTUCE $.29 hd.
TOMATOES $.39 lb.

5.2 CHANGE LEADING ONE-LINE

Format: [wv*x*]

x is the leading, expressed in leading notation (See Chapter 1).

Function: Same as [v command, but effective for one output line.

When Effective: Immediately

When Terminated: End of line; previous stored leading restored.

Example:

```
[wv20][wb]PRODUCE/1<=>
LETTUCE $.29 hd./1<=>
TOMATOES $.39 lb./1<=>
```

PRODUCE

LETTUCE $.29 hd.
TOMATOES $.39 lb.

5.3 CHANGE VERTICAL DIVISION (PROPORTIONAL LEADING)

Format: [vd*x*]

x is a number from 0-100.

Function: This command changes the percentages for proportional leading from those specified in the system standards. The number *x* is the percentage of maximum leading taken from the first line if the leading changes on the next line. (100-*x* is the percentage of maximum lead from the second line). For example, if the system standards call for 50-50 proportional leading, and two consecutive lines call for 12 and 36 point leading, the leading between them will be 24 points. However, if a [vd33] command had overridden the system standards, the leading between the two lines would be 28 points.

If the system standards do not specify proportional leading, this command is ignored.

When Effective: Immediately

When Terminated: By another [vd command.

Example:

```
[p10,12]This is lead before/1<=>
[p24,30]with no proportional leading./1<=>
[p10,12]This is lead before/1<=>
[vd33][p24,30]with proportional leading./1<=>
```

This is lead before

with no proportional leading.

This is lead before

with proportional leading.

5.4 ADDED LEAD

Format: [va*x*]

x is the leading in standard notation (as in [v command).

Function: Adds the specified points of extra leading. Does not affect stored leading. Command is ignored for a hot metal machine.

When Effective: Immediately

Example: `Example of [va7 2]added lead.,/l<->`

Example of

added lead.

5.5 REVERSE LEAD

Format: [vr*x*] or [va-*x*]

x is the leading in standard notation.

Function: Performs reverse lead (film roll-back) by the specified number of points. If the output machine has no reverse lead capability, JUSTIF ignores the command.

When Effective: Immediately

Example: `This week only [vr12]$19.95<=>`

$19.95
This week only

5.6 CANCEL LEADING ONE-LINE

Format: [vz]

Function: Stored leading is not applied to the line containing this command. Does not affect value of stored leading, or reverse and added lead commands.

When Effective: End of line.

When Terminated: Beginning of next line:

Example: `[vz]19.95/1/mXX<=>`

19XX

CHAPTER 6

COLUMN WIDTH CHANGE COMMANDS

6.1 CHANGE COLUMN WIDTH

Format: [c*x*]

x is the column measure in column width notation.

Function: Same command as essential parameter specification. Column width remains in effect until changed. If the specified column width is not permitted in the current setup, JUSTIF ignores the command, issues an error message, and uses the default column width if the specified measure was too small or the maximum column width if the specified measure was too large.

When Effective: At the beginning of a line - effective immediately
In the middle of a line - effective at the beginning of the next line.

When Terminated: By another [c or [cz command.

Example:

```
[c20]The column width change command
is the same as the [c11,6]essential
parameter specification, <->
Once the column width is changed, it
remains until the next C command,<->
```

The column width change command is the same as the essential parameter specification. Once the column width is changed, it remains until the next C command.

6.2 COLUMN RESTORE

Format: [cz]

Function: Cancels all indents and tabs (Chapters 11 and 12). Restores full column width of last [c command.

When Effective: Beginning of next output line.

Example:

```
[c25][l110]This example begins
 with an indent command, which is described
in a later section, [cz]/l<=>
The column restore command changes
the column width back to 25 picas.<=>
```

This example begins with an indent command, which is described in a later section.
The column restore command changes the column width back to 25 picas.

CHAPTER 7

QUAD COMMANDS

7.1 QUAD LEFT

Format: /l

Function: All preceding text on the line is set flush left in column or subcolumn (if tab stops are set), the line is terminated, and leading is applied.

When Effective: Immediately.

Example:

```
Quad left may be used to end a paragraph./l<->
```

Quad left may be used to end a paragraph.

7.2 QUAD RIGHT

Format: /r

Function: All preceding text on the line is set flush right in column or subcolumn (if tab stops are set), the line is terminated, and leading is applied.

When Effective: Immediately

Example:

```
Minimum word spacing is applied <->
on quadded lines./r<->
```

Minimum word spacing is applied on quadded lines.

7.3 QUAD CENTER

Format: /c

Function: All preceding text on the line is set centered in column or subcolumn (if tab stops are set), the line is terminated, and leading is applied.

When Effective: Immediately

Example:
```
[c20]left/lright/rcenter/c<=>
```

left

right

center

7.4 QUAD MIDDLE (SPLIT QUAD)

Format: /q

Function: All preceding text on the line is set flush left in column or subcolumn, but line is not terminated. This allows remaining text on line to be positioned by another quad or tab command (usually /r). If the remaining text cannot fit on the split quad line, the /q command is ignored and the line is justified.

When Effective: Immediately

Example:
```
[c20]<=>
ITEM/qPAGE/r<=>
```

ITEM PAGE

7.5 CONDITIONAL QUAD

Format: /?

Function: Performs /l if no quad command has followed the last text, otherwise has no effect. This command is usually used with macros to execute a quad left if none was issued, but to avoid inserting the extra blank line usually caused by two consecutive quad commands.

When Effective: Immediately

Example:

```
Conditional quad is useful at the <->
end of a macro,/?<->
Macros are described in Chapter 21,/1/?<->
No extra blank here,<->
```

Conditional quad is useful at the end of a macro.
Macros are described in Chapter 21.
No extra blank here.

CHAPTER 8

PARAGRAPH FLUSH AND HANG COMMANDS

8.1 CHANGE PARAGRAPH INDENT

Format: [ip*x*]

x is the indent measure, expressed in standard or alternate (#) indent notation, as described in Chapter 1.

Function: Changes the amount of paragraph indent from the system standard. This command only sets the indent measure; the /p command performs the specified paragraph indent.

When Effective: Immediately

When Terminated: By [ip%] - System standard restored.

Example: See [ip%]

8.2 PARAGRAPH INDENT

Format: /p

Function: Used at the end of a paragraph. Ends the current line with quad left and indents the next line by the amount specified in the last [ip command or the system standard, whichever is in effect.

When Effective: Immediately

Example: See [ip%]

8.3 RESTORE PARAGRAPH INDENT

Format: [ip%]

Function: Cancels any [ip command in effect and restores the system standard paragraph indent. If no [ip command is in effect, this command is ignored.

When Effective: Immediately

Example:

```
The system standard paragraph indent
happens to be zero./pHence this little
paragraph is not indented.[ip3]/p<=>
But this paragraph is indented by three
picas.[ip%]/p<=>
Now the system standard (zero) is restored.<=>
```

The system standard paragraph indent happens to be zero.
Hence this little paragraph is not indented.
        But this paragraph is indented by three picas.
Now the system standard (zero) is restored.

8.4 DEFINE FLUSH AND HANG INDENT

Format: [ih*x*]

x is the indent measure, expressed in standard or alternate (#) indent notation, as described in Chapter 1.

Function: Defines the amount of left indent for flush and hang. This command only sets the indent measure; the /h command executes the flush and hang.

When Effective: Immediately

When Terminated:
1. By [i%]
2. By [cz]
3. By [ih%]

Example: See [ih%]

8.5 FLUSH AND HANG

Format: /h

Function: Used to end a paragraph. This command:

1. Forces a quad left.
2. Causes the next line to be set justified (full out).

3. Causes succeeding lines to be indented left by the measure specified in the previous [ih command.

If no [ih command is in effect, /h functions like /l.

When Effective: Immediately

When Terminated:
1. By another /h (next line set flush, then hang continues).
2. By [ih%]

Example: See [ih%]

8.6 CANCEL FLUSH AND HANG

Format: [ih%]

Function:
1. Terminates flush and hang: performs /l functions on final hanging line; next line set full out.
2. Cancels flush and hang indent specified by previous [ih command.

When Effective: Immediately

Example:

```
[c20]<=>
[ih3]When flush and hang indent is
defined, the flush and hang command
may be used to end a paragraph,/h<=>
The first line of this next paragraph
is then set flush, and the remaining
lines are indented by three picas,/h<=>
The process is repeated each time a
flush and hang<=>
 command is issued, as at the end of
the previous sentence,[ih%]<=>
Now flush and hang is cancelled,/h<=>
Therefore the flush and hang command
is identical to quad left,<=>
```

When flush and hang indent is defined, the flush and hang command may be used to end a paragraph.
The first line of this next paragraph is then set flush, and the remaining lines are indented by three picas.
The process is repeated each time a flush and hang command is issued, as at the end of the previous sentence. Now flush and hang is cancelled.
Therefore the flush and hang command is identical to quad left.

CHAPTER 9

SPACING COMMANDS

These commands all insert a fixed amount of space at the point in text where the command is issued. The character must be defined in FONGEN.

9.1 EM SPACE

Format: /m

Example:
```
This is an EM/mspace,<=>
```

This is an EM space.

9.2 EN SPACE

Format: /n

Example:
```
This is an EN/nspace,<=>
```

This is an EN space.

9.3 THIN SPACE

Format: /i

Example:
```
This is a thin/ispace,<=>
```

This is a thin space.

9.4 FIXED SPACE

Format: /f

Example:
```
This is a fixed/fspace,<=>
```

This is a fixed space.

9.5 FIGURE SPACE

Format: /g

Example:
```
This is a figure/gspace.<=>
```

This is a figure space.

9.6 ONE-UNIT SPACE

Format: /o

Example:
```
This is a one-unit/ospace.<=>
```

This is a one-unitspace.

9.7 THUT (THIN-PLUS)

Format: /;

9.8 BOLT (EN-PLUS)

Format: /:

9.9 EM DASH

Format: —(underline)

Example:
```
[c20]<=>
An EM dash (_) is longer than a  <=>
hyphen (-).<=>
```

An EM dash (—) is longer then a hyphen (-).

CHAPTER 10

PI CASE COMMANDS

The Pi case characters for a particular type font are defined using the FONGEN program. There are three composition commands for calling Pi case characters. One command accesses a single character from Pi case. For accessing several characters, there is a command to enter Pi case used in conjunction with a command to exit Pi case.

10.1 SINGLE PI CHARACTER

Format: +

Function: The single character following + is set from Pi case. Therefore + (like /, [, and]) cannot appear in text as an ordinary character, but must itself be accessed from Pi case or represented by another character. For example, if + is in Pi case, the only command to include + in text is ++.

When Effective: Immediately

Example

```
[c20]<=>
++, +/, +[, and +$ are Pi case <=>
characters.<=>
```

+, /, [, and ¢ are Pi case characters.

10.2 ENTER PI CASE

Format: /+

Function: All characters following this command are accessed from Pi case, until an exit Pi case command is issued.

When Effective: Immediately

When Terminated: By an exit Pi case command:

1. +
2.]
3. /
4. [

Example: See Below

10.3 EXIT PI CASE

Format: +,[,/,or]

Function: Any of these characters cause exit from Pi case, and the character itself is not interpreted as Pi case.

There are two modes of existing Pi case.

1. The + and] characters are not interpreted as command characters.

2. The / and [characters are interpreted as command characters. Thus, a standard composition command can also perform the exit Pi case function.

When Effective: Immediately

Example:

```
[c20]<=>
/+6+ is a Pi cage character.<=>
```

¢ is a Pi case character.

CHAPTER 11

TAB COMMANDS

Four commands are available for setting tab stops. These commands do not position text, but establish the tab stops (similar to a "tab set" key on a typewriter). Tab stops divide a column into subcolumns. **The tab stops set by commands are called explicit tab stops. The left and right margins are always treated as implicit tab stops.** The number of subcolumns in a line is one greater than the number of explicit tab stops. For example, if explicit tab stops are set at 10 and 20 picas in a 30 pica column, there are three subcolumns: between 0 and 10 picas, 10 and 20 picas, and 20 and 30 picas. The maximum number of subcolumns allowed is specified in the system standards for each installation.

Four other commands perform the actual tabulation, using the previously set tab stops. Unlike a typewriter, the tabulation commands:

1. Follow the text to be positioned.
2. Position text within a subcolumn; and
3. Move the carriage to the next tab stop (beginning of the next subcolumn).

11.1 SET TAB STOPS

Format: [ts*x,y,z...*]

x, *y*, and *z* are distances from the left margin, expressed in standard or alternate (#) indent notation.

Function: Set tab stops at the specified distances. The tab stops are effective in addition to tab stops previously set by [ts, [tc, [tp, or /t commands, subject to maximum number of subcolumns.

When Effective: Immediately

When Terminated: 1. By [t%]

2. By [cz]

Example: See Tab Left

11.2 SET EQUAL TAB STOPS

Format: [tc*x*]

x is a number

Function: Sets tab stops for *x* equal subcolumns. For example [tc3] in a 30 pica column sets tab stops at 10 and 20 picas. The tab stops are effective in addition to tab stops previously set by [ts, [tc, or [tp or /t commands, subject to the maximum number of subcolumns.

When Effective: Immediately

When Terminated: 1. By [t%]

2. By [cz]

Example: See Tab Right.

11.3 SET PROPORTIONAL TAB STOPS

Format: [tp*x, y, z,...*]

x, y, and *z* are numbers

Function: Sets tab stops to create subcolumns whose widths are in the ratio *x, y, z,* etc. (By definition, the number of tab stops set is one less than the number of subcolumns specified in this command.) For example, [tp1,3,2] in a 30 pica column sets tab stops for subcolumns in the ratio 1:3:2. In this case the subcolumn widths would be 5, 15, and 10 picas, respectively. The tab stops would be set at 5 and 20 picas. The tab stops are effective in addition to tab stops previously set by [ts, [tc, [tp, or /t commands and the implied tab stops at the column margins, subject to the maximum number of subcolumns.

When Effective: Immediately

When Terminated: 1. By [t%]

2. By [cz]

Example: See Tab Center.

11.4 SET TAB IN TEXT

Format: /t

Function: Sets a tab stop at the point in text where the command is issued. Several /t commands may be issued on a line or on several lines. If the line is justified (instead of tabbed or quadded), the tab stop is set after interword spacing has been adjusted. The tab stop is effective in addition to tab stops previously set by [ts, [tc, [tp or /t commands, subject to the maximum number of subcolumns.

When Effective: Immediately

When Terminated:
1. By [t%]
2. By [cz]

Example: See Tab Around

11.5 TAB LEFT

Format: /u

Function:
1. Positions previous text on the line (since any previous tabulation command) flush left in the current subcolumn.
2. Moves carriage to next tab stop (beginning of next subcolumn.) See Notes and Cautions.

When Effective: Immediately

Example:

```
[c30][ts15,20,25]<=>
Smith/u4/u1/u1/uRobinson/u4/u1/u2/uBrown/u3/u<=>
1/u0/u<=>
```

Smith	4	1	1
Robinson	4	1	2
Brown	3	1	0

11.6 TAB RIGHT

Format: /w

Function:
1. Positions previous text on the line (since any previous tabulation command) flush right in the current subcolumn.
2. Moves carriage to next tab stop (beginning of next subcolumn).

When Effective: Immediately

Example:

```
[c30][tc4]<=>
Smith/u4/w1/w1/wRobinson/u4/w1/w2/wBrown/u3/w<=>
1/w0/w<=>
```

Smith	4	1	1
Robinson	4	1	2
Brown	3	1	0

11.7 TAB CENTER

Format: /v

Function:
1. Positions previous text on the line (since any previous tabulation command) in the center of the current subcolumn.
2. Moves carriage to next stop (beginning of next subcolumn).

When Effective: Immediately

Example:

```
[c30]/l<=>
[tp3,1,1,1]/lPlayer/vab/vr/vh/v<=>
Smith/u4/v1/v1/vRobinson/u4/v1/v2/vBrown/u3/v<=>
1/v0/v<=>
```

Player	ab	r	h
Smith	4	1	1
Robinson	4	1	2
Brown	3	1	0

11.8 TAB AROUND

Format: /a

Function: This command works differently from the other three tabulation commands.

1. The preceding text is centered on the next tab stop.
2. The carriage remains in the next subcolumn.

If issued in the last subcolumn on a line, /a has the effect of /w.

When Effective: Immediately

Example:

```
[c30]<=>
/l/m/mPlay/ter/m/m/m/m/ma/tb/m/m/tr/m/m/th/l<=>
Smith/a4/a1/a1/a/l<=>
```

Player	ab	r	h
Smith	4	1	1

11.9 NOTES AND CAUTIONS FOR USING TABULATION COMMANDS

1. Do not use tabulation commands on justified lines.
2. Use only one /u,/v, or /w per subcolumn.
3. Use only as many /a commands as explicit tab stops.
4. Tab stops are fixed and are not affected by indents, which can interfere with tabulation.
5. A tabulation command in the last column terminates the line.
6. If any subcolumn but the last is overset, the overset tab stop is ignored and further tabulation on the line proceeds to the next tab stop. In other words, oversetting one tab stop displaces subsequent tabs.
7. Do not overset the last subcolumn. This condition causes JUSTIF to attempt to justify the entire line.

11.10 CANCEL TAB STOPS

Format: [t%]

Function: Cancels all tab stops set by [ts, [tc, [tp, and /t commands.

When Effective: Immediately

Example:

```
[c30]/l<=>
[ts10,20]ABC/uDEF/uGHI/u[t%]ABC/uDEF/u<=>
```

```
ABC                    DEF                    GHI
ABC
DEF
```

CHAPTER 12

INDENT COMMANDS

TYPES OF INDENTS:

1. Column indents. These are indents from the left, right, or both column margins. They are specified by measure (standard or alternate indent notation) and remain in effect until cancelled or end-of-take. Fixed (predefined) column indents may also be specified in the system standards and called by commands.

2. Text indents. These are similar to column indents, but are specified by marking a horizontal point in text rather than by measure from the column margin. Text indents require two commands to be executed. One command sets the mark in text for future indents (similar to setting a tab with the /t command). Up to 10 text indent positions are permitted. The actual indent command effects either a left or right indent to a particular mark. Unlike column indents, there is no single text indent command for both left and right indents, but both may be accomplished by two commands. Like column indents, text indents remain in effect until cancelled.

3. Hanging indents. These are indents from the left, right, or both margins which are effective for a specified vertical length. The vertical length is specified in lines and the amount of the indent is specified by measure. Hanging indent commands may also specify a number of lines to wait before beginning the indent. Fixed hanging indents may be predefined in the system standards and called by commands.

4. Length fits. These are left, right, or left-and-right indents which are effective for a specified vertical length, similar to hanging indents. Unlike hanging indents, length fits use a vertical length specified in picas and points, and do not delay before indenting.

5. Skews. These are slanted-margin indents which can set copy around triangles of white space. Skews have effect for a specified vertical length, and specify a maximum indent. A forward skew progressively increases the indent for each line of copy until the maximum indent is reached at the specified vertical distance. (The apex of the "white-space-triangle" is at the top.) A reverse skew begins with the specified maximum indent and progressively decreases the indent for each line of copy until the indent is zero at the specified vertical distance. (The apex of the "white-space-triangle" is at the bottom.) The left, right, or both margins may be skewed, and the six possible skew triangles are schematically shown in Figure 1.

Indent commands of different types are not mutually exclusive. Each type of indent is applied in addition to any other type of indent which may be in effect. Also, indents do not affect tab stops, which remain at absolute positions.

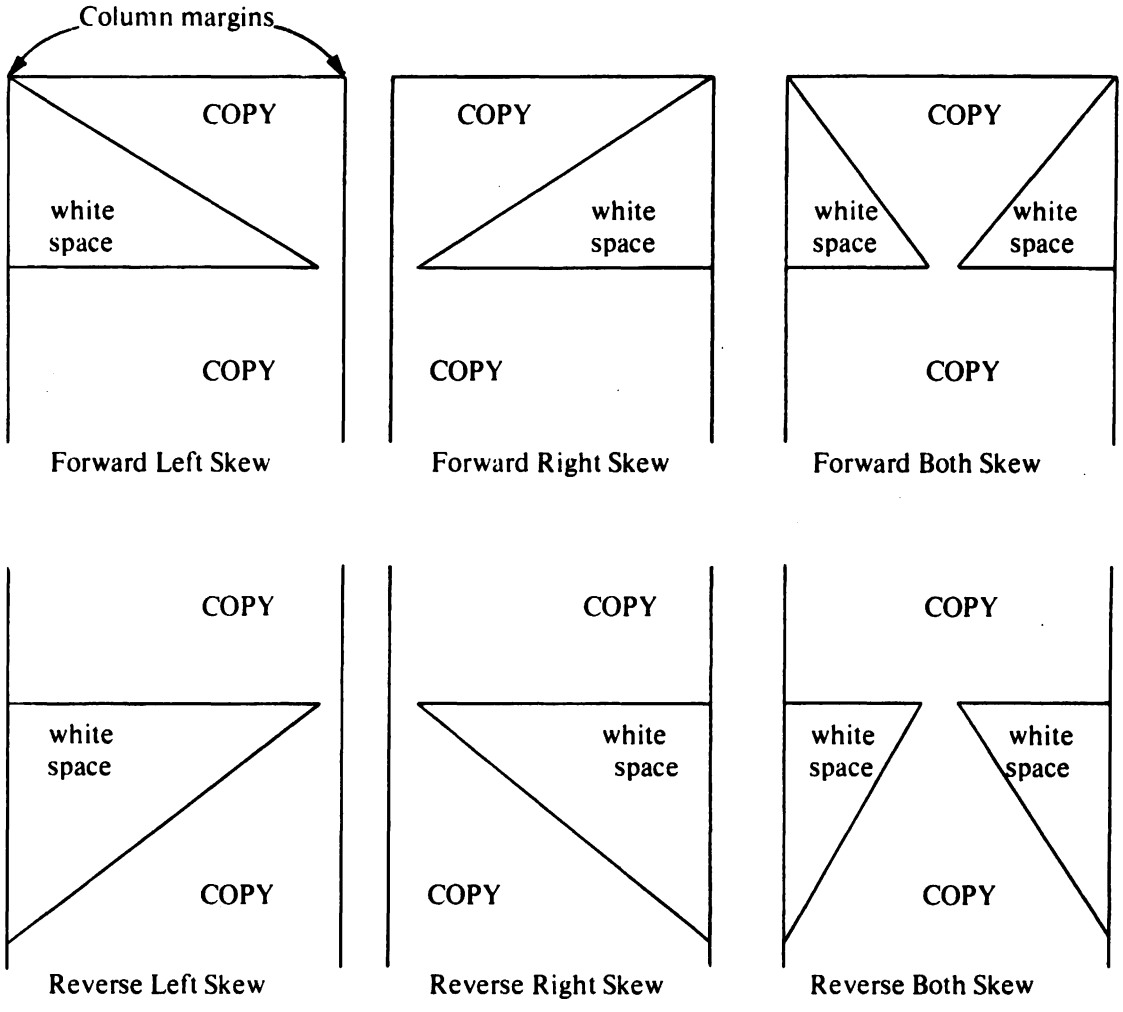

Figure 1.
Types of Skews

12.1 COLUMN INDENT COMMANDS

12.1.1 Indent Left

Format: [il*x*]

x is the measure from the left column margin expressed in standard or alternate indent notation.

Function: Following text is set with the specified left margin indent, until indent command is cancelled or changed.

When Effective:
1. If issued at the beginning of a line (for example after a quad left) — effective immediately.
2. If used in the middle of a line — effective at the beginning of the next line.

When Terminated:
1. By another [il command.
2. By an [ib or [if command.
3. By [il%]
4. By [ib%]
5. By [i%]
6. By [cz]

Example:

```
[c20]A left column indent remains in
effect until cancelled [il5]./l<->
Unless given at the beginning of a
line, the indent begins on the next
line./l[il%]<->
The indent just cancelled was 5 picas
from the left margin.<->
```

A left column indent remains in effect until cancelled.
Unless given at the beginning of a line,
the indent begins on the next line.
The indent just cancelled was 5 picas from the left margin.

12.1.2 Indent Right

Format: [ir*x*]

x is the measure from the right column margin expressed in standard or alternate indent notation.

Function: Following text is set with right margin indent, until indent command is cancelled or changed.

When Effective:
1. If issued at the beginning of a line— effective immediately.
2. If issued in the middle of a line — effective at the beginning of the next line.

When Terminated:
1. By another [ir command.
2. By an [ib or [if command.
3. By [ir%]
4. By [ib%]
5. By [i%]
6. By [cz]

Example:

```
[c20]Right column indents operate the
same as left indents./l<=>
[ir10]This indent command was given
at the beginning of this line./l[ir%l<=>
The indent just cancelled was 10 picas
from the right margin./l<=>
```

Right column indents operate the same as left indents.
This indent command was
given at the beginning of
this line.
The indent just cancelled was 10 picas from the right
margin.

12.1.3 Indent Both

Format: [ib*x*]

x is a measure from both column margins, expressed in standard or alternate indent notation.

Function: Following text is set with the specified indent applied to both margins, until the indent command is cancelled or changed.

When Effective:
1. If issued at the beginning of a line — effective immediately.
2. If issued in the middle of a line — effective at the beginning of next line.

When Terminated:
1. By another [ib command.
2. By an [if command.
3. By an [il command or [il%] (right indent unaffected).
4. By an [ir command or [ir%] (left indent unaffected).
5. By [ib%]
6. By [i%]
7. By [cz]

Example:

```
[c20]The indent both command is <=>
equivalent to giving left and right
column indent commands together,/1[ib5]Like
other column indents, the double indent
persists until cancelled,/1[ib%] <=>
The indent just cancelled was five
Picas from each margin,/1<=>
```

The indent both command is equivalent to giving left and right column indent commands together.

        Like other column indents,
        the double indent persists
        until cancelled.

The indent just cancelled was five picas from each margin.

12.1.4 Fixed Indent

Format: [if*x*]

x is a number from 0-9.

Function: Calls one of the 10 possible fixed indents defined in the system standards (See TSYSCF.MAC Functional Specification). Fixed indents may be defined with both left and right margin indents. If either the

left or right indent is set to zero, the fixed indent is equivalent to either [il or [ir. If both indents are equal,the fixed indent is equivalent to [ib. If the left and right indents are different (the most useful arrangement), the fixed indent is equivalent to a combination of [ir and [il commands, with different indent measures. The fixed indent is effective until changed or cancelled.

When Effective:

1. If issued at the beginning of a line — effective immediately.
2. If issued in the middle of a line — effective at the beginning of next line.

When Terminated:

1. By another [if command.
2. By an [ib command.
3. By an [il command or [il%] (right indent unaffected).
4. By an [ir command or [ir%] (left indent unaffected).
5. By [ib%]
6. By [i%]
7. By [cz]

Example:

```
[c15]Fixed indents are defined in the
system standards./l<->
[1t2]Fixed indents are specified in
points and usually define a double
indent like +[ib, [ib%]/l<->
The fixed indent just cancelled was
12 points from the left margin and
36 points from the right./l<->
```

Fixed indents are defined in the system standards.
    Fixed indents are specified in points and
    usually define a double indent like [ib.
The fixed indent just cancelled was 12 points from the left margin and 36 points from the right.

12.1.5 Cancel Left Indent

Format: [il%]

Function: Cancels left indent set by [ir, [ib, or [if command.

When Effective: Immediately

Example:

```
[c2?]A left column indent remains in
effect until cancelled[!15],/!<->
Unless given at the beginning of a
[!!%]line, the indent begins on the
next line,/!<->
```

A left column indent remains in effect until cancelled.
Unless given at the beginning of a line,
the indent begins on the next line.

12.1.6 Cancel Right Indent

Format: [ir%]

Function: Cancels right indent set by [ir, [ib, or [if command.

When Effective: Immediately

Example: See Indent Right

12.1.7 Cancel Both Indents

Format: [ib%]

Function: Cancels any left indent and any right indent set by [il, [ir, [ib, or [if command.

When Effective: Immediately

Example: See Indent Both

12.2 TEXT INDENT COMMANDS

12.2.1 Define Text Indent

Format: [it*x*]

x is a number from 0—9.

Function: Defines an indent position at the point in text where the command is issued. (On a justified line, the interword expansion is not in-

cluded in determining the horizontal indent position.) Up to 10 text indent positions may be defined to be effective at any time. This command does not perform indents but merely defines the text indent position. The indent position is then available for a left or right text indent, which is called by an [ij or [ik command. Text indent positions remain defined until cancelled or redefined.

When Effective: Immediately

When Terminated:
1. By another [it command using the same indent number.
2. By [i%]
3. By [cz]

Example: See Left Text Indent

12.2.2 Left Text Indent

Format: [ij*x*]

x is an indent position number defined by an [it command.

Function: The following text is indented from the left margin to indent position *x*. The indent remains in effect until cancelled or changed. If no indent position *x* has been defined, JUSTIF issues an error message and ignores the command.

When Effective:
1. If issued at the beginning of a line — effective immediately.
2. If issued in the middle of a line — effective at the beginning of the next line.

When Terminated:
1. By another [ij command.
2. By [ij%]
3. By [i%]
4. By [cz]

Example:

```
[c20][ts5]<->
Format:/u[it1]+[ij[fi]x[fr]+]/1<->
[ij1][fi]x[fr] is an indent position
number defined by an IT command.[ij%]/1/1<->
Function:<->
```

Format: [ij*x*]
x is an indent position number defined by an IT command.

Function:

12.2.3 Right Text Indent

Format: [ik*x*]

x is an indent position number defined by an [it command.

Function: The following text is indented from the right margin to indent position *x*. The indent remains in effect until cancelled or changed. If no indent position *x* has been defined, JUSTIF issues an error message and ignores the command.

When Effective:
1. If issued at the beginning of a line — effective immediately.
2. If issued in the middle of a line — effective at the beginning of the next line.

When Terminated:
1. By another [ik command.
2. By [ik%]
3. By [i%]
4. By [cz]

Example:

```
[c20]<->
This line sets a text indent mark[it2]./1<->
[ik2]A right text indent works like
a left indent. The IT command sets
horizontal marks in text for both types
of indents./1[ik%]<->
The indent persists until cancelled,
as illustrated by this line full out
to the right./1<->
```

This line sets a text indent mark.
A right text indent works like a
left indent. The IT command
sets horizontal marks in text for
both types of indents.
The indent persists until cancelled, as illustrated by
this line full out to the right.

12.2.4 Cancel Left Text Indent

Format: [ij%]

Function: Cancels the [ij command currently in effect, if any (the indent position remains defined).

When Effective: Immediately

Example: See Left Text Indent

12.2.5 Cancel Right Text Indent

Format: [ik%]

Function: Cancels the [ik command currently in effect, if any. (The indent position remains defined.)

When Effective: Immediately

Example: See Right Indent

12.3 HANGING INDENT COMMANDS

12.3.1 Left Hanging Indent

Format: [hl*x,y,z*]

x is the indent from the left margin, expressed in standard or alternate indent notation.

y is the number of lines to wait before the indent takes effect; *y* may be zero.

z is the number of lines to hang; *z* may be zero to specify an indent which continues until cancelled.

A zero *y* value may be specified by omitting a number and writing only a comma. A zero *z* value may be specified by omitting both the number and preceding comma.

Function: The indent begins after *y* lines have been set. Then text is indented the specified measure from the left margin. The indent continues until *z* lines have been set or until cancelled.

When Effective: As specified by *y*.

If the command is issued at the beginning of a line, that line itself is included in the count of lines to wait. If issued in the middle of a line, the count begins on the following line.

When Terminated:

1. After *z* lines have been indented ($z = 0$)
2. By [hl%]
3. By another [hl or [hb command.
4. By [hb%]
5. By [i%]
6. By [cz]

Examples:

```
[c20][hl10,1,3]<->
This is a left hanging indent of ten
picas which begins after one line.
The hang lasts for three lines, after
which the indent is cancelled and the
full column width is restored. Hanging
indents are useful for vertical copy
fitting, as in picture runarounds./l<->
```

This is a left hanging indent of ten picas which begins after one line. The hang lasts for three lines, after which the indent is cancelled and the full column width is restored. Hanging indents are useful for vertical copy fitting, as in picture runarounds.

```
[c20][hl10]<->
This is a left hanging indent of ten
picas in which both the y and z values
are omitted. The zero y value means
that the hang begins immediately.[hl%]The
zero z value means that the hang lasts
until cancelled, as shown at the beginning
of this sentence./l<->
```

This is a left hanging indent of ten picas in which both the y and z values are omitted. The zero y value means that the hang begins immediately.The zero z value means that the hang lasts until cancelled, as shown at the beginning of this sentence.

12.3.2 Right Hanging Indent

Format: [hr*x,y,z*]

x is the indent from the right margin expressed in standard or alternate indent notation.

y and *z* are as described for [hl command.

Function: The indent begins after *y* lines have been set. Then text is indented the specified measure from the right margin. The indent continues until *z* lines have been set or until cancelled.

When Effective: Same as [hl command.

When Terminated:

1. After *z* lines have been indented ($z = 0$).
2. By [hr%]
3. By another [hr or [hb command.
4. By [hb%]
5. By [i%]
6. By [cz]

Example:

```
[c20][hr10,,3]<->
This is a right hanging indent of ten
picas. The hang lasts for three lines.
The difference between this example
and the first left hanging indent example
is that the number of lines to wait
before the hang is zero. After three
lines the column width is restored./l<->
```

This is a right hanging in-
dent of ten picas. The hang
lasts for three lines. The
difference between this example and the first left hanging indent example is that the number of lines to wait before the hang is zero. After three lines the column width is restored.

12.3.3 Hanging Indent Both

Format: [hb*x,y,z*]

x is the indent from both left and right margins; *y* and *z* are as described for [hl command.

Function: Both margins are indented by the same measure. Equivalent to [hl and [hr commands in effect simultaneously (each specifying the same indent).

When Effective: Same as [hl command.

When Terminated:

1. After *z* lines have been indented ($z = 0$).
2. By [hb%]
3. By an [hl command or [hl%] - right indent unaffected.
4. By an [hr command or [hr%] - left indent unaffected.
5. By [i%]
6. By [cz]

Example:

```
[c20][hb5,1,3]<->
This is a hanging indent on both margins.
One line is set full out before the
indent begins. [hr%] Now the right
portion of the indent has been cancelled
but the left indent continues until
the three lines have been set, Then
the normal column width is restored.<->
```

This is a hanging indent on both margins. One line is
set full out before the in-
dent begins.Now the right
portion of the indent has been cancelled
but the left indent continues until the three lines have been set. Then the normal column width is restored.

12.3.4 Fixed Hanging Indent

Format: [hf*x*]

x is a number from 0-9.

Function: Calls one of the 10 possible fixed (predefined) hanging indents, which are established by the system standards. A fixed hanging indent functions like an [hb, except that the left and right indent measures may be different. The left and right indents (either of which may be zero), the delay parameter, and the line count parameter are all defined in the system standards. If the indent number is undefined, the command is ignored.

When Effective: Same as [hl command.

When Terminated: Same as [hb command.

12.3.5 Cancel Left Hanging Indent

Format: [hl%]

Function: Cancels any left hanging indent set by [hl, [hb, or [hf command.

When Effective: Immediately

Example: See [hl command.

12.3.6 Cancel Right Hanging Indent

Format: [hr%]

Function: Cancels any right hanging indent set by [hr, [hb, or [hf command.

When Effective: Immediately

Example: See [hb command.

12.3.7 Cancel Both Hanging Indents

Format: [hb%]

Function: Cancels all hanging indents: [hb, [hf, [hl, and [hr

When Effective: Immediately

Example:

```
[c20][hb5]<->
This 5 pica hang begins immediately
and is applied to both margins. The
command specifies an indefinite hang./1<->
[hb%]Now the hang is cancelled, as
illustrated by this sentence.<->
```

This five pica hang begins immediately and is applied to both margins. The command specifies an indefinite hang.
Now the hang is cancelled, as illustrated by this sentence.

12.4 LENGTH FIT COMMANDS

12.4.1 Left Length Fit

Format: [11*x*,*y*]

x is the indent from the left margin, expressed in standard or alternate indent notation. *y* is the vertical length of the indent, expressed in picas or picas and points, separated by a period.

Function: Following text is indented the specified measure from the left margin. The indent continues for the specified vertical measure or until cancelled.

When Effective:
1. If issued at the beginning of a line — effective immediately
2. If issued in the middle of a line — effective at the beginning of the next line.

When Terminated:
1. After the vertical length specified.
2. By [11%]

3. By another [ll command or [lb command.
4. By [lb%]
5. By [i%]
6. By [cz]

Example:

```
[p12,14][c20]Length fits will be longer
than specified if the vertical length
cannot be achieved exactly using the
stored leading.[ll#10,6.6]<->
This is a left length fit of 10 ens.
The length fits lasts for a vertical
distance of six and one-half picas
(six picas and six points). Length
fits are similar to hanging indents.
After the specified vertical distance,
the original column width is restored
and lines are set full out. The length
fit described above has ended with
the previous line./l<->
```

Length fits will be longer than specified if the vertical length cannot be achieved exactly using the stored leading.This is a left length fit of 10 ens. The length fits lasts for a vertical distance of six and one-half picas (six picas and six points). Length fits are similar to hanging indents. After the specified vertical distance, the original column width is restored and lines are set full out. The length fit described above has ended with the previous line.

12.4.2 Right Length Fit

Format: [lr*x*,*y*]

x is the indent from the right margin, expressed in standard or alternate indent notation.

y is the vertical length of the indent, expressed in picas or picas and points, separated by a period.

Function: Same as [ll, except text is indented from right margin.

When Effective: Same as [ll command.

When Terminated:
1. After the vertical length specified.
2. By [lr%]
3. By another [lr or [lb command.
4. By [lb%]
5. By [i%]
6. By [cz]

Example:

```
[c20]Length fits will be longer
than specified if the vertical length
cannot be achieved exactly using the
stored leading.[lr3,3]<->
This is a right length fit of 3 picas
which lasts for a vertical distance
of 3 picas. It works the same as a
left length fit. Another important
difference between hanging indents
and length fits is that the vertical
distance is specified in lines for
a hanging indent, but by measure for
a length fit./i<->
```

Length fits will be longer than specified if the vertical length cannot be achieved exactly using the stored leading.This is a right length fit of 3 picas which lasts for a vertical distance of 3 picas. It works the same as a left length fit. Another important difference between hanging indents and length fits is that the vertical distance is specified in lines for a hanging indent, but by measure for a length fit.

12.4.3 Length Fit Both

Format: [lb*x,y*]

x is the indent measure, expressed as in [ll or [lr.

y is the vertical length of the indent, as in [ll or [lr.

Function: Both margins are indented by the same measure. Equivalent to [ll and [lr command in effect simultaneously.

When Effective: Same as [ll.

When Terminated:

1. After the vertical length specified.
2. By [lb%]
3. By another [lb command.
4. By an [ll command or [ll%] — right indent unaffected.
5. By an [lr command or [lr%] — left indent unaffected.
6. By [i%]
7. By [cz]

Example:

```
[p12,14][c20]<->
Length fits will be longer than specified
if the vertical length cannot be achieved
exactly using the stored leading.
[lb3,4]<->
This is a length fit on both margins
of 3 picas which lasts for a vertical
distance of four picas. It works like
a left and right length fit together.
Note that unlike hanging indents, length
fits cannot specify a delay before
indenting./l<->
[lb%]<->
```

Length fits will be longer than specified if the vertical length cannot be achieved exactly using the stored leading. This is a length fit on

> both margins of 3 picas which lasts for a vertical distance of four picas. It works like a left and right length fit together.

Note that unlike hanging indents, length fits cannot specify a delay before indenting.

12.4.4 Cancel Left Length Fit

Format [ll%]

Function: Cancels any left length fit set by [ll or [lb.

When Effective: Immediately

Example:

```
[p12,14][c20]<->
Length fits will be longer than specified
if the vertical length cannot be achieved
exactly using the stored leading.<->
[ll6,60]<->
This is a left length fit set up for
a 6 pica indent and a vertical length
of 60 picas. [ll%]<->
However, the length fit was cancelled
at the beginning of this sentence./l<->
```

Length fits will be longer than specified if the vertical length cannot be achieved exactly using the stored leading.This is a left length fit set up for a 6 pica indent and a vertical length of 60 picas. However, the length fit was cancelled at the beginning of this sentence.

12.4.5 Cancel Right Length Fit

Format: [lr%]

Function: Cancels any right length fit set by [lr or [lb.

When Effective: Immediately

Example:

```
[p12,14][c20]<->
Length fits will be longer than specified
if the vertical length cannot be achieved
exactly using the stored leading.<->
[lr6,60]<->
This is a right length fit set up for
a 6 pica indent and a vertical length
of 60 picas. [lr%]<->
However, the length fit was cancelled
at the beginning of this sentence./l<->
```

Length fits will be longer than specified if the vertical length cannot be achieved exactly using the stored leading.This is a right length fit set up for a 6 pica indent and a vertical length of 60 picas. However, the length fit was cancelled at the beginning of this sentence.

12.4.6 Cancel Length Fit Both

Format: [lb%]

Function: Cancels all length fits: [ll, [lr, and [lb.

When Effective: Immediately

Example:

```
[p12,14][c20]<->
Length fits will be longer than specified
if the vertical length cannot be achieved
exactly using the stored leading.<->
[lb3,60]<->
This is a length fit set up for a 3
pica indent on both margins and a vertical
length of 60 picas. [lb%]<->
However, the length fit was cancelled
at the beginning of this sentence./l<->
```

Length fits will be longer than specified if the vertical length cannot be achieved exactly using the stored leading.This is a length fit set up for a 3 pica indent on both margins and a vertical length of 60 picas. However, the length fit was cancelled at the beginning of this sentence.

12.5 SKEW COMMANDS

12.5.1 Forward Left Skew

Format: [sl*x,y*]

x is the maximum left indent, expressed in standard or alternate indent notation.
y is the vertical length of the skew, expressed in picas or picas and points, separated by a decimal point.

Function: Following this command, the first line of text is justified across the column. Each following line is left indented by a progressively greater amount, so that the specified maximum indent is attained at the specified vertical distance. Then succeeding lines are justified as usual. A left skew may be cancelled or redefined before the vertical length has been reached.

When Effective:

1. If issued at the beginning of a line — effective immediately.
2. If issued in the middle of a line — effective at the beginning of the next line.

When Terminated: After the specified vertical length.

2. By [sl%]
3. By another [sl command or an [sb command.
4. By [sb%]
5. By [i%]
6. By [cz]

Example:

```
[c20][sl10,7]<->
This is a forward left skew. Each line
is left indented slightly more than
the line before. The vertical length
of the skew is seven picas. After this
distance, the maximum indent of ten
picas is achieved. Then the remaining
lines of type are set full out, as
illustrated by this sentence which
extends across the column./l<->
[sl10,7]<->
This is the same skew in effect again.
However, now the same skew is cancelled
before the vertical length is reached.
[sl%]<->
This illustrates the cancel left skew
command./l<->
```

This is a forward left skew. Each line is left indented slightly more than the line before. The vertical length of the skew is seven picas. After this distance, the maximum indent of ten picas is achieved. Then the remaining lines of type are set full out, as illustrated by this sentence which extends across the column.
This is the same skew in effect again. However, now the same skew is cancelled before the vertical length is reached.
This illustrates the cancel left skew command.

12.5.2 Reverse Left Skew

Format: [sl-*x*,*y*]

x is the maximum left indent, expressed in standard or alternate indent notation.

y is the vertical length of the skew; expressed in picas or picas and points, separated by a period.

Function: Following this command, the first line is left indented by the maximum amount specified in the command. Each following line is left indented by a progressively lesser amount, so that the last line is jus-

tified across the column at the specified vertical distance. Then succeeding lines are justified as usual. The command may be cancelled or redefined before the vertical length has been reached.

When Effective: Same as [sl command.

When Terminated: Same as [sl command.

Example:

```
[c20][sl-10,7]<->
This is a reverse left skew. The first
line is indented to the maximum indent
of ten picas. Then each line is left
indented slightly less than the line
before. The vertical length of the
skew is seven picas. After this distance,
the indent is reduced to zero. The
remaining lines of type are set full
out./l<->
```

This is a reverse left skew. The first line is indented to the maximum indent of ten picas. Then each line is left indented slightly less than the line before. The vertical length of the skew is seven picas. After this distance, the indent is reduced to zero. The remaining lines of type are set full out.

12.5.3 Forward Right Skew

Format: [sr*x*,*y*]

x is the maximum right indent, expressed in standard or alternate indent notation.

y is the vertical length of the skew expressed in picas or picas and points, separated by a period.

Function: Same as [sl, except that text is indented from right margin. May be cancelled or redefined before the vertical length has been reached.

When Effective: Same as [sl command.

When Terminated:
1. After the specified vertical length.
2. By [sr%]

3. By another [sr or an [sb command.

4. By [sb%]

5. By [i%]

6. By [cz]

Example:

```
[c20][sr10,7]<->
This is a forward right skew. Each
line is right indented slightly more
than the line before. The vertical
length of the skew is seven picas.
After this distance, the maximum indent
of ten picas is achieved. Then the
remaining lines of type are set full
out as illustrated by this sentence
which extends across the column. /l<->
[sr10,7]<->
This is the same skew in effect again.<->
 However, now the skew is cancelled
before the vertical length is reached.
[sr%]<->
This illustrates the Cancel Right Skew
Command./l<->
```

This is a forward right skew. Each line is right indented slightly more than the line before. The vertical length of the skew is seven picas. After this distance, the maximum indent of ten picas is achieved. Then the remaining lines of type are set full out, as illustrated by this sentence which extends across the column.
This is the same skew in effect again. However, now the skew is cancelled before the vertical length is reached.This illustrates the Cancel Right Skew Command.

12.5.4 Reverse Right Skew

Format: [sr-*x*,*y*]

x is the maximum right indent, expressed in standard or alternate indent notation

y is the vertical length of the skew, expressed in picas or picas and points, separated by a period.

Function: Same as [sl- (reverse left skew), except that text is indented from right margin. May be cancelled or redefined before the vertical length has been reached.

When Effective: Same as [sl.

When Terminated: Same as [sl.

Example:

```
[p12,14][c20][sr-10,7]<->
This is a reverse right skew.  The
first line is indented to the maximum
indent of ten picas.  Then each line
is right indented slightly less than
the line before.  The vertical length
of the skew is seven picas.  After
this distance, the indent is reduced
to zero.  The remaining lines of type
are set full out./1<->
```

This is a reverse right skew. The first line is indented to the maximum indent of ten picas. Then each line is right indented slightly less than the line before. The vertical length of the skew is seven picas. After this distance, the indent is reduced to zero. The remaining lines of type are set full out.

12.5.5 Forward Skew Both

Format: [sb*x,y*]

x is the maximum indent (left and right margins), expressed in standard or alternate indent notation.

y is the vertical length of the skew, expressed in picas or picas and points, separated by a period.

Function: Both margins indented by the same measure. Equivalent to an [sl or [sr command, with identical indents and vertical lengths. May be cancelled or redefined before vertical length is reached.

When Effective: Same as [sl.

When Terminated:

1. After the specified vertical length.
2. By [sb%]
3. By another [sb command
4. By an [sl command or [sl%] — right skew unaffected.
5. By an [sr command or [sr%] — left skew unaffected.
6. By [i%]
7. By [cz]

Example:

```
[c20][p12,14][sb5,7]<->
This is a forward skew on both left
and right margins. Each line is indented
slightly more than the line before.
The vertical length of the skew is
seven picas. After this distance, the
maximum indent of five picas per margin
is achieved. Then the remaining lines
of type are set full out, as illustrated
by this sentence.<->
[sb5,7]<->
This is the same skew in effect again.
However, now the skew is cancelled
before the vertical length is reached.
[sb%]<->
This illustrates the Cancel Skew Both
Command./1<->
```

This is a forward skew on both left and right margins. Each line is indented slightly more than the line before. The vertical length of the skew is seven picas. After this distance, the maximum indent of five picas per margin is achieved. Then the remaining lines of type are set full out, as illustrated by this sentence.This is the same skew in effect again. However, now the skew is cancelled before the vertical length is reached. This illustrates the Cancel Skew Both Command.

12.5.6 Reverse Skew Both

Format: [sb-*x*,*y*]

Function: Both margins indented by the same measure. Equivalent to an [sl- and [sr- command, with identical indents and vertical lengths. May be cancelled or redefined before vertical length is reached.

When Effective: Same as [sl.

When Terminated: Same as [sb.

Example:

```
[c2%][p12,14][sb-5,7]<~>
This is a reverse skew on both left
and right margins. The first line is
indented to the maximum indent of five
picas. Then each line is indented slightly
less than the line before. The vertical
length of the skew is seven picas.
After this distance, the indent is
reduced to zero. The remaining lines
of type are set full out./!<~>
```

This is a reverse skew on both left and right margins. The first line is indented to the maximum indent of five picas. Then each line is indented slightly less than the line before. The vertical length of the skew is seven picas. After this distance, the indent is reduced to zero. The remaining lines of type are set full out.

12.5.7 Cancel Left Skew

Format: [sl%]

Function: Cancels any left skew set by [sl or [sb.

When Effective: Immediately

Example: See Forward Left Skew

12.5.8 Cancel Right Skew

Format: [sr%]

Function: Cancels any right skew set by [sr or [sb.

When Effective: Immediately

Example: See Forward Right Skew.

12.5.9 Cancel Skew Both

Format: [sb%]

Function: Cancels any left and/or right skew set by [sl, [sr, or [sb.

When Effective: Immediately

Example: See Forward Skew Both.

12.5.10 Cancel All Indents Command

Format: [i%]

Function: Cancels any indent described in this section.

When Effective: Immediately

Example:

```
[p10,12][c20]<->
The five [!t1] types of indents are
additive./!<->
[!]1][!]4][h!4[!!2,12[s!-2,12]<->
The indent applied to this line is
the sum of all the indents in the
command at the beginning of this sentence,
[!%]<->
Now all indents are cancelled and the
previous column width of 20 picas is
used./!<->
```

The five types of indents are additive.

.in 10
The indent applied to this line is the sum of all the indents in the command at the beginning of this sentence. Now all indents are cancelled and the previous column width of 20 picas is used.

CHAPTER 13

MARK POINT COMMANDS

Mark points are vertical tabs defined in text. Up to 10 mark points may be defined via command. These mark points may then be used in commands for vertical carriage positioning and for setting copy in columns. Mark point commands may be used if the output photocomposition machine has reverse lead capability and if mark point commands are allowed in the system standards.

13.1 SET VERTICAL MARK

Format:	[vm *x*]
	x is the mark point number (0-9).
Function:	Defines mark point *x* at the current vertical position. The mark point remains defined until cancelled or redefined.
When Effective:	Immediately
When Terminated:	1. By another [vm command using the same mark point number.
	2. By [vm*x*%]
	3. By [vm%]
Example:	See Vertical Position.

13.2 CANCEL VERTICAL MARK

Format:	[vm*x*%]
Function:	Cancels the definition of mark point *x*.
When Effective:	Immediately
Example:	See Vertical Position.

13.3 CANCEL ALL MARKS

Format: [vm%]

Function: Cancels all existing mark points.

When Effective: Immediately

Example: See [vc.

13.4 VERTICAL POSITION

Format: [vp*x*]

x is a mark point number.

Function: Reverse or forward lead to mark point *x*. The carriage is not returned to left column margin. If no mark point *x* is defined, the command is ignored.

When Effective: Immediately

Example:

```
[o20]<->
[vm1]Sm[th/|<->
Robinson/|<->
Brown/|<->
[vp1][[|4]4/|<->
4/|<->
3/|<->
[vp1][[|0]1/|<->
0/|<->
2/|<->
[vm1x][vp1]Does this return?/|<->
```

```
Smith     4    1
Robinson  4    0
Brown     3    2
                Does this return?
```

13.5 VERTICAL COLUMNS

Format: [vc*x*,*y*]

x is the vertical column length (in points)
y is a mark point number

Function: This command causes columns to be set across the paper of a photocomposition machine. The column width is the width currently in effect, i.e., the width specified in the last [c command, the space between the columns (gutter) is either the gutter width specified in the last [vg command, or the default gutter width defined in the system standards (if no [vg command was used).

The first column begins at the left margin. After *x* points of lead (the vertical column length specified in the [vc command), an auto column return to mark point *y* is performed. That is, the carriage is moved horizontally by the gutter width and vertically back to mark point *y*.

If no mark point *y* is defined, the vertical position of the [vc command itself defines mark point *y*.

Then another column of length *x* is set next to the first column, and another return to mark point *y* is made after horizontal gutter width spacing. When no more columns can fit across the paper, the next auto column return is made to the left margin under the leftmost column and an extra blank line is inserted. Mark point *y* is redefined as this vertical position. Columns continue to be set until cancelled.

When Effective: Immediately

When Terminated: By [vc%]

Example:

```
[vm%][c11]vg2][vc72,1]<=>
First all previous marks are cancelled,
so the [vc command defines mark point
1 at the vertical position where the
command is given.  This example sets
three columns across the paper.  Each
column is 11 picas wide, the gutter
width between the columns is specified
to be 2 picas, and the paper width
is 45 picas.  If the column widths
were set to 20 picas, only two columns
would be set across the paper.  The
length of each column is 72 points.
This example ends with a Cancel Vertical
Columns command, which returns to the
left margin.[vc%]/1<=>
Columns are now cancelled./1<=>
```

First all previous marks are cancelled, so the [vc command defines mark point 1 at the vertical position where the command is given. This example sets three columns across the paper. Each column is 11 picas wide, the gutter width between the columns is specified to be 2 picas, and paper width is 45 picas. If the column widths were set to 20 picas, only two columns would be set across the paper. The length of each column is 72 points. This example ends with a Cancel Vertical Columns command,

which returns to the left margin.

Columns are now cancelled.

13.6 CANCEL VERTICAL COLUMNS

Format: [vc%]

Function: Terminates column setting function of [vc. Returns carriage under leftmost column and inserts a blank line. If no [vc command is in effect, this command is ignored.

When Effective: Immediately

Example: See [vc.

13.7 VERTICAL BOTH

Format: [vb]

Function: This command is used when a [vc command is in effect and allows manual instead of automatic column returns, combining functions of [vp and [vc. When this command is issued, an immediate column return is made to the vertical position of the last mark point, without waiting for leading. If no mark point is defined when [vb] is issued, the command is ignored.

When Effective: End of current line.

Example:

```
[vm%][c1[vg2[vc72,1]<->
This column ends with a normal return
via the +[vb command.[vb]/|<->
Now we should be in the second column,
and columns continue to be set by
the +[vc command, until cancelled./|<->
The +[vc command is an extremely useful
mark point feature. To use mark point
commands, the output machine must perform
reverse lead, and mark point must be
allowed in the system standards./|[vc%]<->
```

This column ends with a normal return via the [vb command.

Now we should be in the second column, and columns continue to be set by the [vc command, until cancelled.
The [vc command is an extremely useful mark point feature. To use mark point commands, the output machine must perform reverse lead, and mark point must be allowed in the system standards.

13.8 VERTICAL GUTTER

Format: [vg*x*]

x is the gutter width, expressed in standard or alternate indent notation.

Function: Overrides the system standard default gutter width. Superseded by another [vg command.

When Effective: Immediately

Example: See [vc.

CHAPTER 14

FRACTION COMMANDS

These commands insert the appropriate fraction characters at the point in text where the command is issued.

14.1 ONE-EIGHTH (1/8)

Format: /1

Example:

```
[c20]<=>
You can fool all the people /1 of the
time,<=>
```

You can fool all the people ⅛ of the time.

14.2 ONE-FOURTH (1/4)

Format: /2

Example:

```
[c20]<=>
You can fool all the people /2 of the
time,<=>
```

You can fool all the people ¼ of the time.

14.3 THREE-EIGHTHS (3/8)

Format: /3

Example:

```
[c20]<=>
You can fool /3 of the people /2 of
the time,<=>
```

You can fool ⅜ of the people ¼ of the time.

14.4 ONE-HALF (1/2)

Format: /4

Example: See /8

14.5 FIVE-EIGHTHS (5/8)

Format: /5

Example:

```
[c20]<=>
You can't fool /5 of the people /1
of the time.<=>
```

You can't fool ⅝ of the people ⅛ of the time.

14.6 THREE-FOURTHS (3/4)

Format: /6

Example:

```
[c20]<=>
You can fool /6 of the people some
of the time.<=>
```

You can fool ¾ of the people some of the time.

14.7 SEVEN-EIGHTHS (7/8)

Format: /7

Example:

```
[c20]<=>
You can fool /7 of the people all of
the time.<=>
```

You can fool ⅞ of the people all of the time.

14.8 ONE-THIRD (1/3)

Format: /8

Example:
```
[c20]<=>
You can fool /8 of the people /4 of
the time.<=>
```

You can fool ⅛ of the people ¼ of the time.

14.9 TWO-THIRDS (2/3)

Format: /9

Example:
```
[c20]<=>
You can't fool all the people /9 of
the time.<=>
```

You can't fool all the people ⅔ of the time.

CHAPTER 15

RAGGED MODE COMMANDS

Ragged mode commands allow unjustified lines of copy to be set. Ragged lines may be set flush left (ragged right), flush right (ragged left), or centered on some line within the column (ragged on both sides). Ragged lines have minimum word spacing and no letter spacing. Hyphenation is performed unless specifically disabled by [ya%]. Only one ragged mode command may be in effect at any time; if another is issued it supersedes the previous command.

15.1 RAGGED LEFT

Format: [rl]

Function: Subsequent lines set flush right. Ragged mode continues until cancelled or superseded.

When Effective:
1. If issued at the beginning of a line — effective immediately.
2. If issued in the middle of a line — effective at the beginning of next line.

When Terminated:
1. By [r%]
2. If superseded by [rr], [rc], or [rm]

Example:

```
[c15][rl]<->
Ragged mode commands allow unjustified
lines of copy to be set.  Ragged lines
may be set flush left (ragged right),
flush right (ragged left), or centered
on some line within the column. <->
Ragged lines have minimum word spacing
and no letter spacing./l[r%] <->
Ragged mode continues until cancelled
as at the beginning of this sentence./l<->
[va12]<->
```

Ragged mode commands allow unjustified lines of copy to be set. Ragged lines may be set flush left (ragged right), flush right (ragged left), or centered on some line within the column. Ragged lines have minimum word spacing and no letter spacing.
Ragged mode continues until cancelled as at the beginning of this sentence.

15.2 RAGGED RIGHT

Format: [rr]

Function: Subsequent lines set flush left. Ragged mode continues until cancelled or superseded.

When Effective: Same as [rl.

When Terminated: 1. By [r%]

2. If superseded by [rl], [rc] or [rm]

Example:

```
[c15][rr]<=>
Ragged mode commands allow unjustified
lines of copy to be set.  These lines
are set ragged right.  Ragged mode continues
until cancelled or superseded./1<=>
[va12]<=>
```

Ragged mode commands allow unjustified lines of copy to be set. These lines are set ragged right. Ragged mode continues until cancelled or superseded.

15.3 RAGGED CENTER

Format: [rc]

Function: Subsequent lines have both ragged margins. Each line is centered in the column. Ragged mode continues until cancelled or superseded.

When Effective: Same as [rl.

When Terminated: 1. By [r%]

2. If superseded by [rl], [rr], or [rm]

Example:

```
[c15][rc]<=>
Ragged mode commands allow unjustified
lines of copy to be set.  These lines
are set ragged center.  Each line is
centered on the column midpoint.  Both
left and right margins are ragged./1<=>
[va12]<=>
```

Ragged mode commands allow unjustified lines of copy to be set. These lines are set ragged center. Each line is centered on the column midpoint. Both left and right margins are ragged.

15.4 RAGGED MIDDLE

Format: [rm] or [rm*x*]

x is a horizontal measure, expressed in points.

Function: Subsequent lines have both ragged margins. Lines are staggered as follows:

The first line is centered about a position *x* points to the left of the column center. The second line is centered about a position *x* points to the right of the column center, and so on. If the measure *x* is not specified in the command, the default ragged middle offset defined in the system standards is used.

When Effective: Same as [rl.

When Terminated: 1. By [r%]

2. If superseded by [rl], [rr], or [rc]

Example:

```
[c15][rm24]<=>
Ragged mode commands allow unjustified
lines of copy to be set. These lines
are set according to a ragged middle
command. Ragged middle is similar to
ragged center because both margins
are ragged. However, ragged center
lines are centered on the column midpoint.
Each ragged middle line is centered
on a point offset from the column midpoint,
alternating to the right and left
of the midpoint. In this example, the
horizontal offset from the midpoint
is 24 points [rs]/1<=>
```

Ragged mode commands allow unjustified lines of copy to be set. These lines are set according to a ragged middle command. Ragged middle is similar to ragged center because both margins are ragged. However, ragged center lines are centered on the column midpoint. Each ragged middle line is centered on a point offset from the column midpoint, alternating to the right and left of the midpoint. In this example, the horizontal offset from the midpoint is 24 points

15.5 CANCEL RAGGED MODE

Format:	[r%
Function:	Cancels any ragged mode command in effect. If none is in effect, the command is ignored.
When Effective:	Same as [rl.
Example:	See [rl.

CHAPTER 16

OUTPUT COMMANDS

16.1 ENABLE AUTO RULE DROP

Format: [or]

Function: This command allows automatic rule drop to occur after certain quad commands. When auto rule drop is enabled, JUSTIF automatically outputs a rule across the column after a line terminated by a quad command. The particular quad commands which can activate auto rule drop are specified in the system standards. For example, one installation may specify rule drop after quad left or quad right, another after quad left or quad center. Auto rule drop remains enabled for a take, unless cancelled.

When Effective: Immediately

When Terminated: By [or%]

Example:

```
[c11][p6,6[or]<_>
Complete set of 1954 New York Giants
baseball cards, with duplicates of
Monte Irvin and Wes Westrum. Every
boy's dream, $37,000, 634-5789./1<_>
[or%]Situation wanted/1<_>
Mark-up man, good work habits, moderate
drinker./1<_>
```

Complete set of 1954 New York Giants baseball cards, with duplicates of Monte Irvin and Wes Westrum. Every boy's dream, $37,000, 634-5789.

Situation Wanted
Mark-up man, good work habits, moderate drinker.

16.2 CANCEL AUTO RULE DROP

Format: [or%]

Function: Disables auto rule drop.

When Effective: Immediately

Example: See [or.

16.3 CANCEL FLASH

Format: [of%]

Function: Cancels flashing of characters, but allows carriage to move. Used for copy-fitting, for example, setting tabs in text before the text itself is output, and arranging display copy with varying point sizes. Flashing remains disabled until restored. If flash is already cancelled, the command is ignored.

When Effective: Immediately

When Terminated: By [of]

Example:

```
[c20][of%]Robinson/m/m/t[of]/1
Smith/u4/1<=>
Robinson/u4/1[v.100]<=>
```

Smith 4
Robinson 4

```
[c30][p36,18[of%]Bananas/m/m/n/t[p18[of]15¢<=>
/1[p36]Bananas/u[p18]1b/1[va100]<=>
```

Bananas 15¢
lb

16.4 RESTORE FLASH

Format: [of]

Function: Restores flashing of characters after an [of%] command. If flash is already on, the command is ignored.

When Effective: Immediately

When Terminated: By [of%]

Example: See [of%]

16.5 CANCEL FLASH AND LEAD (ONE-LINE)

Format: [ob]

Function: Cancels both flash and stored leading for one output line only. Has similar use to [of%] command. This command must be issued at the beginning of a line. Flash and leading remain cancelled until the end of line.

When Effective: Immediately.

When Terminated: End of line.

Example:

```
[c30[p18,20][ob]Tender/m/m/t/l<->
Young/l<->
Tender/u[p36]Chickens/q39+$ lb./r<->
```

Young
Tender **Chickens 39¢ lb.**

16.6 SELECT OUTPUT DEVICE

Format: [o*x*]

x is a device number. See ALLOT Manual for details.

Function: Usually the output typesetting device is selected from the setup by ALLOT. This command overrides the selection by ALLOT in order to force output to a particular device or punch.

This command is not generally used.

When Effective: Immediately

Example: [02]

16.7 SET DELAY

Format: [op*x*]

x is a number of tape feeds.

Function: This command sets the number of tape feeds used for a delay on an output tape for a hot metal machine. If no [op command is used in a take, the system standard default for tape feeds is used. Tape feed delays are issued automatically when required by hot metal drivers,

and may be issued on demand via the [od] command. The number of tape feeds set by the [op command supersedes the system standard for the rest of the take, unless cancelled or reset by another [op.

When Effective: Immediately.

When Terminated: 1. By [op%]

2. By another [op command.

Example: See [od].

16.8 RESET DELAY

Format: [op%]

Function: Cancels [op command; restores the system standard number of tape feeds per delay. If no [op is in effect, the command is ignored.

When Effective: Immediately

Example: See [od].

16.9 OUTPUT DELAY

Format: [od]

Function: Outputs the number of tape feeds specified by the current [op command or the system standard, whichever is in effect.

When Effective: Immediately

Example: `####/c[op75][od][op%]<↵>`

`####`

CHAPTER 17

LEADER COMMANDS

Leaders are character sequences which may be used instead of white space to fill subcolumns on certain tabbed or quadded lines.

Standard leaders are dots (...). Alternate leaders may also be defined for use instead of standard leaders. Leader mode may be activated for all tabbed or quadded lines (with either standard or other leaders). One-shot leaders (standard or other) may also be used. Either standard or other leader mode, but not both, may be in effect at one time.

17.1 USE LEADERS

Format: [xl]

Function: Subsequent tabbed or quadded lines are set with standard leaders instead of white space. Standard leader mode continues until cancelled or superseded by other leader mode.

When Effective: Immediately

When Terminated: 1. By[x%]

2. By [xx]

Example:

```
[c20][xL]Weather/a41/r[x%]Guide to features/c<->
```

Weather.....41
Guide to features

17.2 DEFINE OTHER LEADERS

Format: [xdl]

l is the character sequence to be used as other leaders. The sequence may be up to six characters. Single Pi characters may be included in the other leader sequence, and the + commands used to access these

Pi characters do not count towards the limit of six. Fixed space characters may also be included.

This command must include the terminating right square bracket.

Function: Defines the character sequence to be used as other leaders. This command only defines the other leader characters and does not request other leaders (This is accomplished by [xx] and /x commands). The definition of other leaders remains in effect until superseded by another [xd command.

When Effective: Immediately.

When Terminated: By another [xd command.

Example: See [xx]

17.3 USE OTHER LEADERS

Format: [xx]

Function: Subsequent tabbed or quadded lines are set with other leaders instead of white space. The other leader characters must be defined in an [xd command. If none are defined, JUSTIF issues an error message and ignores the command. If a whole number of other leader sequences do not fill the space on the line, white space is inserted on the left. Other leader mode continues until cancelled or superseded by standard leader mode.

When Effective: Immediately

When Terminated: 1. By [x%]

2. By [xl]

Example:

```
[c20][xd_][xx]Amusements/q31/rArts/q40/r<->
```

Amusements----------31
Arts----------40

17.4 CANCEL LEADER MODE

Format: [x%]

Function: Cancels standard [xl] or alternate [xx] leader mode, whichever is in effect. If neither is in effect, the command is ignored.

When Effective: Immediately

Example: See [xl]

17.5 ONE-SHOT LEADERS

Format: /.

Function: Sets the current line or subcolumn using standard leaders instead of white space. Must be issued before the quad command which ends the line or the tab command which ends the subcolumn. If standard leader mode is already in effect as a result of [xl], this command is ignored.

When Effective: When the line or subcolumn is set.

Example:

```
[c20]Amusements/q/.31/r<->
```

Amusements..... 31

17.6 ONE-SHOT OTHER LEADERS

Format: /x

Function: Sets the current line or subcolumn using other leaders instead of white space. Must be issued before the quad command which ends the line or the tab command which ends the subcolumn. The other leader characters must be defined in an [xd command. If none are defined, JUSTIF issues an error message and ignores the command. If a whole number of other leader sequences do not fill the space on the line, white space is inserted on the left. If other leader mode is already in effect as a result of [xx], this command is ignored.

When Effective: When the line or subcolumn is set.

Example:

```
[c20]<->
[xd*]Sweepstakes/q/x50/r<->
```

Sweepstakes **********50

17.7 ONE-SHOT RULE

Format: /—

Function: Sets the current line or subcolumn using a rule instead of white space. Must be issued before the quad command which ends the line or the tab command which ends the subcolumn.

When Effective: When the line or subcolumn is set.

Example:

```
[c20]Arts/q/_50/r<=>
```

Arts_____50

17.8 ONE-SHOT WHITE SPACE

Format: /s

Function: The reverse of /. and /x. In either standard or alternate leader mode, this command sets the current line or subcolumn using white space instead of leaders. If neither leader mode is in effect, this command is ignored. Must be issued before the quad command which ends the line or the tab command which ends the subcolumn.

When Effective: When the line or subcolumn is set.

Example:

```
[c20][xL]Rain Today/c<=>
Guide to Features/s/c<=>
Arts/q41/r<=>
```

..... Rain Today .....
Guide to Features
Arts..... 41

CHAPTER 18

JUSTIFICATION COMMANDS

18.1 JUSTIFICATION PARAMETERS

Format: [jp*w*,*x*,*y*,*z*]

w, *x*, *y*, and *z* are numbers (relative units).

Function: Overrides system standards for remainder of take, or until superseded by another [jp command.

w = minimum word space

x = maximum word space before hyphenation attempted

y = maximum word space before letter spacing attempted

z = maximum letter space

All parameters are in relative units.
If any parameter is omitted, the system standard default is preserved.

When Effective: Immediately

When Terminated: By another [jp command.

Example:

```
[c11[p9,9[jp8,17,32,6]/1<->
Typeset-10 Composition Commands control
the appearance of typeset copy and
select the set-up for alloting copy
to a typesetting machine.  Composition
commands are included within the text
of copy, which may be input to Typeset-10
via the editing programs COPYED and
PRUFED, or the wire service translator
WIPTIN./1<->
[jp8,48,32,6]<->
Typeset-10 Composition Commands control
the appearance of typeset copy and
select the set-up for alloting copy
to a typesetting machine.  Composition
commands are included within the text
of copy, which may be input to Typeset-10
```

```
via the editing programs COPYED and
PRUFED, or the wire service translator
WIPTIN./1<->
```

Typeset-10 Composition Commands control the appearance of typeset copy and select the set-up for alloting copy to a typesetting machine. Composition commands are included within the text of copy, which may be input to Typeset-10 via the editing programs COPYED and PRUFED, or the wire service translator WIPTIN.

Typeset-10 Composition Commands control the appearance of typeset copy and select the set-up for alloting copy to a typesetting machine. Composition commands are included within the text of copy, which may be input to Typeset-10 via the editing programs COPYED and PRUFED, or the wire service translator WIPTIN.

18.2 ALLOW LETTER SPACING

Format: [jl*x*]

x is a number (relative units)

Function:

1. Enables letter spacing for remainder of take or until cancelled.
2. Overrides the system standard default for minimum letter space to *x* (relative units). Minimum letter space is usually zero or one for cold type and six relative units for hot metal. The amount of letter space remains in effect until the end of take, unless redefined by another [jl command.

When Effective: When the current line is justified.

When Terminated:

1. By [jl%] (disable)
2. By another [jl command. (amount of space).

Example:

```
[c11[p9,9[jp8,8,8,6[jl%]/1<->
Typeset-10 Composition Commands control
the appearance of typeset copy and
select the set-up for alloting copy
to a typesetting machine. Composition
```

```
commands are included within the text
of copy, which may be input to Typeset-10
via the editing programs COPYED and
PRUFED, or the wire service translator
WIPTIN,/1<->
[j11]<->
Typeset-10 Composition Commands control
the appearance of typeset copy and
select the set-up for alloting copy
to a typesetting machine.  Composition
commands are included within the text
of copy, which may be input to Typeset-10
via the editing programs COPYED and
PRUFED, or the wire service translator
WIPTIN,/1<->
```

Typeset-10 Composition Commands control the appearance of typeset copy and select the set-up for alloting copy to a typesetting machine. Composition commands are included within the text of copy, which may be input to Typeset-10 via the editing programs COPYED and PRUFED, or the wire service translator WIPTIN.

Typeset-10 Composition Commands control the appearance of typeset copy and select the set-up for alloting copy to a typesetting machine. Composition commands are included within the text of copy, which may be input to Typeset-10 via the editing programs COPYED and PRUFED, or the wire service translator WIPTIN.

18.3 CANCEL LETTER SPACING

Format: [jl%]

Function: Disables letter spacing until end of take unless reinstituted by a [jl command. If letter spacing is already disabled, the command is ignored.

When Effective: When the current line is justified.

When Terminated: By a [jl command.

Example: See [jl.

18.4 ONE-LINE JUSTIFY

Format: /j

Function: All preceding text on the current line is justified across the column. The current line is terminated; copy following this command is set beginning on the next line.

When Effective: Immediately

Example:

```
[c20[p12,14[jp6,32,48,6]<=>
This line is forced to be justified./j<=>
After the one-line justify command,
text is justified as usual, as illustrated
by this sentence./l<=>
[va12]<=>
```

This line is forced to be justified. After the one-line justify command, text is justified as usual, as illustrated by this sentence.

CHAPTER 19

HYPHENATION COMMANDS

19.1 HYPHENATION PARAMETERS

Format: [yp*w,x,y,z*]

w,x,y, and *z* are numbers.

Function: Overrides system standards for remainder of take, or until superseded by another [yp command.

w = minimum number of letters before a hyphen.

x = minimum number of letters after a hyphen.

y = maximum number of successive hyphens allowed.

z = minimum number of letters in word to be hyphenated.

If any parameter is omitted, the system standard default is preserved. If the allowable successive end-of-line hyphens (y) is specified as zero, hyphenation is disabled.

When Effective: Immediately

When Terminated: By another [yp command.

Example:

```
[p9,9[jp6,12,48,6][yp2,2,4,4]<=>
Typeset-10 Composition Commands control
the appearance of typeset copy and
select the set-up for alloting copy
to a typesetting machine, Composition
commands are included within the text
of copy, which may be input to Typeset-10
via the input+/editing programs COPYED
and PRUFED or the wire service translator
WIPTIN,/1<=>
```

```
[c11[p9,9[j1%[jp6,12,48,6][yp2,2,1,4]<->
Typeset-10 Composition Commands control
the appearance of typeset copy and select
the set-up for alloting copy to a typesetting<->
 machine. Composition commands are
included within the text of copy, which
may be input to Typeset-10 via the
input+/editing programs COPYED and
PRUFED or the wire service translator
WIPTIN./1<->
```

Typeset-10 Composition Commands control the appearance of typeset copy and select the set-up for alloting copy to a typesetting machine. Composition commands are included within the text of copy, which may be input to Typeset-10 via the input/editing programs COPYED and PRUFED or the wire service translator WIPTIN.

Typeset-10 Composition Commands control the appearance of typeset copy and select the set-up for alloting copy to a typesetting machine. Composition commands are included within the text of copy, which may be input to Typeset-10 via the input/editing programs COPYED and PRUFED or the wire service translator WIPTIN.

19.2 DISABLE HYPHENATION

Format:	[ya%]
Function:	Disables hyphenation until end of take, unless later enabled by [ya]. If hyphenation is already disabled, the command is ignored.
When Effective:	Immediately
When Terminated:	By [ya]

Example:

```
[c11[p9,9[jp8,12,32,6[ya%]/1<=>
Typeset-10 Composition Commands control
the appearance of typeset copy and
select the set-up for alloting copy
to a typesetting machine.  Composition
commands are included within the text
of copy, which may be input to Typeset-10
via the editing programs COPYED and
PRUFED, or the wire service translator
WIPTIN./1<=>
[ya]<=>
Typeset-10 Composition Commands control
the appearance of typeset copy and
select the set-up for alloting copy
to a typesetting machine.  Composition
commands are included within the text
of copy, which may be input to Typeset-10
via the editing programs COPYED and
PRUFED, or the wire service translator
WIPTIN./1<=>
```

Typeset-10 Composition Commands control the appearance of typeset copy and select the set-up for alloting copy to a typesetting machine. Composition commands are included within the text of copy, which may be input to Typeset-10 via the editing programs COPYED and PRUFED, or the wire service translator WIPTIN.
Typeset-10 Composition Commands control the appearance of typeset copy and select the set-up for alloting copy to a typesetting machine. Composition commands are included within the text of copy, which may be input to Typeset-10 via the editing programs COPYED and PRUFED, or the wire service translator WIPTIN.

19.3 ENABLE HYPHENATION

Format: [ya]

Function: Enables hyphenation if previously disabled by [yp or [ya%] command. If hyphenation is already enabled, the command is ignored.

When Effective: Immediately

When Terminated:
1. By [ya%]
2. By [ypn,n,0,n]

Example: See [ya%]

19.4 ONE-WORD DISABLE

Format: [yw]

Function: Disables hyphenation for the single word following this command only. Hyphenation is automatically reinstituted after one word. A word is considered terminated by:

1. any space
2. punctuation marks except period and comma
3. hyphen or dash in text
4. a tab or quad command

This command must immediately precede the word for which hyphenation is disabled, with no intervening space.

When Effective: Immediately

When Terminated: End of word

Example:

```
[c10,5[p9,9[jp8,12,32,6]/1<=>
Typeset-10 Composition Commands control
the appearance of typeset copy and
select the set-up for alloting copy
to a typesetting machine. Composition
commands are included within the text
of copy, which may be input to Typeset-10
via the input+/editing programs COPYED and
PRUFED, or the wire service translator
WIPTIN,/1<=>
[jp9,48,32,6]<=>
Typeset-10 Composition Commands control
the appearance of typeset copy and
select the set-up for alloting copy
to a typesetting machine. Composition
commands are included within the text
of copy, which may be input to Typeset-10
via the input+/editing programs COPYED and
PRUFED, or the wire service translator
[yw]WIPTIN,/1<=>
```

Typeset-10 Composition Commands control the appearance of typeset copy and select the set-up for alloting copy to a typesetting machine. Composition commands are included within the text of copy, which may be input to Typeset-10 via the input/editing programs COPYED and PRUFED, or the wire service translator WIPTIN.

Typeset-10 Composition Commands control the appearance of typeset copy and select the set-up for alloting copy to a typesetting machine. Composition commands are included within the text of copy, which may be input to Typeset-10 via the input/editing programs COPYED and PRUFED, or the wire service translator WIPTIN.

19.5 USE DICTIONARY ONLY

Format: [yd]

Function: Only the exception word dictionary is used to hyphenate all words until cancelled. Ordinarily, JUSTIF first consults the dictionary for allowable hyphenation, then uses a hyphenation algorithm if the word does not appear.

When Effective: Immediately

When Terminated: By [ya]

19.6 DISCRETIONARY HYPHEN

Format: /-

Function: This command is used within a word to define allowable hyphenation breaks. If the word must be hyphenated to justify a line, a discretionary hyphen overrides both the exception word dictionary and the hyphenation logic. In other words, this command either forces a particular hyphenation of a word or permits none, if the discretionary hyphenation would not permit the desired justification.

When Effective: If the word containing the discretionary hyphen is hyphenated.

Example:

```
[p12,14][c13]Final completion of the
proj-<->
ect<->
```

Final completion of the proj-
ect

CHAPTER 20

MESSAGE COMMANDS

These commands output messages. They are used to punch visual messages on output paper tape, set tag lines on film, or both. The terminating right square bracket is required in all formats.

20.1 TAG LINE MESSAGE

Format: [mt*x*]

x is the text of any message

Function: The message (*x*) is set on film, using current face and point size at the time the command is effective. The message is preceded and followed by blank lines.

When Effective: At the next quad command.

Example:

```
[o11[o9,10]<=>
Further announcements are expected
today.[mtHOLD THIS TAKE]/| <=>
In related news, rumors of bread riots were
denied by the city fathers./|/|[va50]<=>
```

Further announcements are expected today.

HOLD THIS TAKE

In related news, rumors of bread riots were denied by the city fathers.

20.2 PUNCH MESSAGE

Format: [mp*x*]

x is the text of any message.

Function: Punches the visual message on the output paper tape. If the typesetting machine is online and no tape is punched, the message is printed at the log device.

When Effective: Immediately

Example:

```
[o11[p9,10]<=>
Further announcements are expected
today.[mpHOLD THIS TAKE]/|<=>
```

Further announcements are expected today.

20.3 BOTH MESSAGES

Format: [mb*x*]

x is the text of any message.

Function: Combination of [mt and [mb commands. The message is set as a tag line and punched as a visual. The tag line is set as described for [mt.

When Effective:

1. Tag line - same as [mt.
2. Punch visual - same as [mp.

CHAPTER 21

MACROS

21.1 WHAT IS A MACRO?

A macro (sometimes called a "format" in computer typesetting) is a sequence of composition commands (and possibly text) which is defined by mark-up personnel and given a name. Then, the contents of the macro may be inserted repeatedly in a take by calling the name of the macro within the text. These two steps, defining and later calling a macro, are accomplished by special composition commands and are described in detail in this section.

More complex macros may be defined with several parts, so that copy from text may be merged with composition commands from the macro, part by part. In other words, after a macro call, some text may be processed, then commands or text from the macro definition inserted, then more text, then a second group of commands or text from the macro, and so on.

21.2 WHEN TO USE MACROS

Macros are used to set up general copy layouts, which are used repeatedly. The macro definition may be set up for merging text. The macro may be called many times, merging different text with the same composition commands each time. This type of macro is useful for display advertising copy, such as grocery ads, where the same layout is repeated with different copy. Another frequent use of macros is for defining complicated layouts, so that the mark-up need not be repeated if the layouts are used again. Another use is repeated insertion of the same text.

21.3 HOW TO DEFINE A MACRO

The first time a macro is defined, the definition is stored on disk as a DECsystem-10 file in the user's area (according to project-programmer number). The filename is the macro name in the definition and the extension is .MCR. Once a macro is stored, it can be called by the same user in later takes without redefinition. If another macro is defined using the same name in the same user area, the second definition overrides the first in storage. The system manager can manually create a system storage area for public (universally available) macro definition files.

A macro definition has the form:

```
[*macroname=body=]
```

1. The definition begins with a left square bracket.

2. The asterisk identifies a macro command.

3. The macro name is from one to six characters.[1]

4. The contents or body of the macro, enclosed in equal signs, is the sequence of composition commands and/or text.

5. The definition must end with a right square bracket.

The macro definition is stored when the macro is defined.

21.4 HOW TO CALL A MACRO

A macro call has the form:

[*macroname]

1. The macro call also begins with a left square bracket and asterisk.

2. The macro name is the same used in the definition.

3. The macro call also ends with a right square bracket.

Examples of Simple Macro Definitions and Calls

This macro just inserts text:

```
[*ed=[o20][p18[fb]FINAL EDITION/oLATE
STOCKS/c=]<->
[*ed]<->
```

FINAL EDITION
LATE STOCKS

This macro inserts essential parameters at the beginning of a take:

```
[*set1=[d401[f401[p10,12[c11]=]<->
[*set1]<->
```

[1]The following characters are not allowed in macro names:
+,/,=,«,[,], or space.

21.5 MACROS WITH SEVERAL PARTS: MERGING TEXT

Macros with several parts are defined using the form:

[*macroname=part1 « part2« ...« partn=]

In the macro definition, the parts of the macro body are separated by the symbol «. When the macro is called, the call is followed by pieces of text to be merged. The pieces of text are called arguments, and are also separated by « symbols. Following the macro call, the first part of the macro body (up to the first « in the definition) is inserted in the output. Then the first argument (up to the first « in text) is merged into the output. Then the second part of the macro body (up to the second « in the definition) is inserted. Then the second argument (up to the second « in text) is merged, and so on. Usually, the number of arguments is equal to the number of parts in the macro definition. Stated another way, the number of « symbols following a macro call is usually equal to the number of « symbols in the definition.

Example of Macro Definitions and Calls for Merge Text

Here's a simple grocery layout macro.

```
[*g1=[c30[p36,18[fb[of%]@/m/m/m/t[p18[of]@/l<->
[p36]@/u[p18]@/l/l=]<->
[*g1]BANANAS@15¢$@BANANAS@LB@<->
[*g1]ORANGES@79¢$@ORANGES@BAG@<->
```

BANANAS **15¢ LB**

ORANGES **79¢ BAG**

Definition:[*gl=[c30[p36,18[fb[of%]
« /m/m/m/t[p18[of]« /l[p36]« /u[p18]« /l/l=]

call: [*g1]BANANAS« 15+$« BANANAS« lb«

Call	Definition	Gives
[*g1]	[c30[p36,18[fb[of%]	[c30[p36,18[fb[of%]
BANANAS	«	BANANAS
«	/m/m/m/t[p18[of]	/m/m/m/t[p18[of]
15+$	«	15+$
«	/l[p36]	/l[p36]
BANANAS	«	BANANAS
«	/u[p18]	/u[p18]
lb	«	lb
«	/l/l	/l/l

Result:[c30[p36,18[fb[of%]BANANAS/m/m/m/t[p18[of]15+$/l[p36]BANANAS/u[p18]lb/l/l

Here's a refinement of the macro, which defines another macro inside it so that the item name does not have to be typed twice. (This is not the same as a nested macro described later in this section.)

```
[o30]/|<=>
[*g2=[c30[p36,18[fb[of%][*|tem=@=]<=>
[*|tem]/m/m/m/t[p18[of]@/|[p36][*|te@]/u[p18]<=>
@/|/|=]<=>
[*g2]APPLES@7¢S@EACH@<=>
```

APPLES 7¢ EACH

Here's a more complicated macro example.

```
[*g3=[c15[p6,7]@/|[ob[p18[fb][*|tem=@=]<=>
[*pr|ce=@=][*|tem]/t[p8]/q/,/t[p18][*pr|ce]/<=>
r<=>
[p6,65]/u[fr]@/v/|/u@/v/|/|[fb[p18][*|te=][p<=>
8]/q/.[p18][*pr|ce]/r/|[fr+t%]=]<=>
[*g3]GERMAN CHOCOLATE@CAKE@$1.35@Fresh@Good@<=>
```

GERMAN CHOCOLATE

CAKE Fresh Good **$1.35**

Using Fewer Arguments

Suppose a macro is defined with several parts to merge several arguments, but the mark-up man wishes to terminate the macro call after merging fewer arguments. The Macro Kill command is used to terminate the macro call. The format of the command is

[ak]

Given as an argument in text, the Macro Kill command terminates macro processing even if more parts remain in the definition. Processing is continued with the text following the Macro Kill command. The [ak] command may not be used within a macro definition.

Macro Kill Example

```
[o30]/|<=>
[*g4=[o30[fb[p36]@[fr][p18]/q/,@@/r/↑/|=]/|<=>
[*g4]BANANAS@15+S@LB@<=>
[*g4]WATERMELON@89+S[ak]/r/|/|[*g4]ORANGES@79<=>
+S@BAG@<=>
```

BANANAS .....15¢LB
WATERMELON.....89¢
ORANGES.....79¢BAG

21.6 REPEATING A MACRO CALL

There are two ways to repeat a macro call.

1. The Macro Repeat command in text,
2. An extra @.

The macro repeat command has the format

[arn]

Where n is a number greater than 0.

Macro Repeat causes the next macro call to be repeated the specified number of times. If the macro accepts arguments, n sets of arguments must follow the call. The [ar command may not be used within a macro definition.

Example of Macro Repeat Command

```
[ar2][*g3]Rainbow@TROUT@$1.69 Lb@Fresh@<->
Caught@Rib@ROAST@$1.89 Lb@USDA@CHOICE@<->
```

Rainbow
TROUT Fresh Caught **$1.69 Lb**
Rib
ROAST USDA CHOICE **$1.89 Lb**

An @ will recall the last macro used, provided the macro call was not terminated by [ak].

Example of @ Macro Recall

```
[*g3]<->
Maine@LOBSTER@$2.25/nLb@TODAY@O.Lyar resn.
Gro<->
und@HAMBURG@.89/nLb@extra@Lean@<->
```

Maine

TODAY ONLY

LOBSTER.....$2.25 Lb

Fresh Ground

Extra Lean

HAMBURG.....89 Lb

21.7 NESTED MACROS

A macro definition may include a call to another macro. Such macros are described as nested. The definition which includes the macro call is referred to as the top level macro, and the called macro is referred to as the second level macro. In turn, the definition of this second level macro may include calls to other macros. Such macros are nested a third level with respect to the original macro definition.

Macros may be nested to a depth of seven levels. If the macros take arguments, each macro has its arguments honored in turn as it is called. For example, suppose macro 1 has three arguments, and macro 2 is nested in macro 1 after the first argument. Macro 2 has two arguments. The order of arguments following the call to macro 1 would be:

1. First argument — macro 1
2. First argument — macro 2
3. Second argument — macro 2
4. Second argument — macro 1
5. Third argument — macro 1

Example of Nested Macros

```
[*descr=[f1[p10,12]/u@/v/1/u@/v=]<->
[*g5=/1/1/1/1[fr[t%[c30[p12,12]@/1[va24[vz[p36[FB][*item=@=<->
][*price=@=][*item]/t[p14]/q/,/t[p36][*price]<->
/r[vr24][*descr]=]<->
[*g5]German Chocolate@CAKE@$1.35@Fresh@Good@<->
```

German Chocolate

Fresh Good

CAKE.....$1.35

Terminating Nested Macros: Macro Kill Again

Another form of the Macro Kill command may be used for terminating nested macros. The form is

[akn]

Where n is the number of levels terminated.

For example, the argument [ak1] terminates the current level without merging further arguments and returns to the previous level. The argument [ak2] terminates two levels and returns to the third level up. All levels are terminated by [ak9].

When a macro is terminated with an [akn] command, it can be recalled later by @.

Example of Terminating a Nested Macro

```
[*q5]German Chocolate@CAKE@$1.35@Fresh@[ak1]<->
[*q5]WHITE@@READ@30¢@$@Pound@Loaf@<->
```

German Chocolate

CAKE *Fresh* **.....$1.35**

WHITE

BREAD *Pound Loaf* **.....30¢**

CHAPTER 22

ARITHMETIC COMMANDS

The Typeset-10 arithmetic commands manipulate integers (whole numbers). The commands use registers or "storage boxes", which are recognized by JUSTIF. Registers may be given single-letter (A to Z) names by the user; therefore 26 registers are available. A register is undefined until it is loaded with a specific integer or the contents of another register (which is also an integer). The basic arithmetic command operations are:

1. Loading a register
2. Adding, multiplying, or dividing the contents of a register with an integer (which may come from another register).
3. Incrementing (adding one to) a register.
4. Displaying a register (inserting the contents of a register into the output of JUSTIF).

22.1 COMMAND SYNTAX

All arithmetic commands begin with a left square bracket, the letter z and a second letter. These are followed, depending on the command, by a register name, an integer, two register names, or one register name and an integer. (Two arguments are separated by a comma.) All arithmetic commands must end with a right square bracket; they may not be concatenated.

Arithmetic commands work from right to left; the first register named receives the result of the arithmetic operation.

Descriptions and examples of all arithmetic commands follow.

22.2 LOAD REGISTER

Format:	[zx*r,s*]
	r is a register name (a-z) *s* may be an integer or a register if previously defined (loaded).
Function:	The value of *s* is loaded into register *r*.
Example:	See next example

22.3 DISPLAY REGISTER (ARABIC)

Format: [za*s*]

s is a register name or integer

Function: The value of *s* is inserted into the output in the form of an arabic number.

Example:
```
[zxa,5][zaa]/l<_>
[zxb,6][zxa,b][zaa]/l<=>
```

5
6

22.4 DISPLAY REGISTER (WORDS)

Format: [zw*s*]

s is a register name or integer

Function: The value of *s* is inserted into the output in the form of upper-case words.

Example:
```
[zxb,3][zwb]<=>
```

THREE

22.5 ZERO REGISTER

Format: [zz*r*]

r is a register name

Function: Register *r* is loaded with zero.

Example:
```
[zzc][zac]<=>
```

0

22.6 INCREMENT REGISTER

Format: [zi*r*]

r is a register name (must be previously defined)

Function: One is added to register *r*.

Example:
```
[zxa,1][zia][zaa]<=>
2
```

22.7 DISPLAY AND INCREMENT REGISTER

Format: [zj*r*]

r is a register name (must be previously defined)

Function: The contents of register *r* is inserted into the output in the form of an arabic number. Then one is added to register *r*.

Example:
```
[zxb,7][zjb]/l[zab]<=>
7
8
```

22.8 ADD TO REGISTER

Format: [zp*r,s*]

r is a register name
s is a register name or integer

Function: The value of *s* is added to register *r* and the result is stored in register *r*.

Example:
```
[zxb,30][zxa,2][zpa,b][zaa]/l[zab]/l<=>
32
30
```

22.9 MULTIPLY REGISTER

Format: [zm*r,s*]

r is a register name
s is a register name or integer

Function: The value of *s* is multiplied by register *r* and the result is stored in register *r*.

Example:
```
[zxc,5][zxd,5][zmc,d][zaq]/|<->
```
25

22.10 DIVIDE REGISTER

Format: [zd*r,s*]

r is a register name
s is a register name or integer

Function: The value of *r* is divided by the value of *s*.
The integer result is stored in *r*.

Example:
```
[zxq,18][zxr,6][zdq,r][zaq]/l<->
```
3

22.11 REMAINDER

Format: [zc*r,s*]

r is a register name
s is a register name or integer

Function: The value of *r* is divided by the value of *s*. The remainder is stored in *r*.

Example:
```
[zxq,20][zxr,6][zcq,r][zaq]/l<->
[zxm,100][zcm,2][zam]/l<->
```
2
0

22.12 USING ARITHMETIC COMMANDS

The two main uses of arithmetic commands are:

1. Performing computations and displaying the result in JUSTIF output.

2. Performing computation and using the result in another type of composition command. Any composition command which requires a numeric argument will accept the contents of an arithmetic register instead. This is indicated by writing the register name, preceded by an exclamation mark, in the command in place of the numeric value. For example, if arithmetic commands are used to compute a point size value in

register a, the point size may then be changed by the command [p!a]. Registers may also be used for alternate indent notation. For example, an indent number of ems may be computed and stored in register z, then the command [il#!z] will execute the indent.

Examples: Here are two macros to print folios.

```
[*folios=[FB][z%a,1][*folio=Typeset-10
Composition Commands/mPage/m[zja]/rPage/m[zja<->
]/mTypeset-10 Composition Commands/l=]=]<->
[*folios][ar8][*folio]<->
```

Typeset-10 Composition Commands Page 1

Page 2 Typeset-10 Composition Commands

Typeset-10 Composition Commands Page 3

Page 4 Typeset-10 Composition Commands

Typeset-10 Composition Commands Page 5

Page 6 Typeset-10 Composition Commands

Typeset-10 Composition Commands Page 7

Page 8 Typeset-10 Composition Commands

Typeset-10 Composition Commands Page 9

Page 10 Typeset-10 Composition Commands

Typeset-10 Composition Commands Page 11

Page 12 Typeset-10 Composition Commands

Typeset-10 Composition Commands Page 13

Page 14 Typeset-10 Composition Commands

Typeset-10 Composition Commands Page 15

Page 16 Typeset-10 Composition Commands

```
[*folios=[FB][z%a,1][*title=@=]=]<->
[*folio=[*title]/mPage/m[zwa][zia]/r<->
Page/m[zwa][zia]/m[*title]/l=]<->
[*folios]Typeset-10 Installation Guide@[ar8][*<->
folio]<->
```

Typeset-10 Installation Guide Page ONE

Page TWO Typeset-10 Installation Guide

Typeset-10 Installation Guide Page THREE

Page FOUR Typeset-10 Installation Guide

Typeset-10 Installation Guide Page FIVE

Page SIX Typeset-10 Installation Guide

Typeset-10 Installation Guide Page SEVEN

Page EIGHT Typeset-10 Installation Guide

Typeset-10 Installation Guide Page NINE

Page TEN Typeset-10 Installation Guide

Typeset-10 Installation Guide Page ELEVEN

Page TWELVE Typeset-10 Installation Guide

Typeset-10 Installation Guide Page THIRTEEN

Page FOURTEEN Typeset-10 Installation Guide

Typeset-10 Installation Guide Page FIFTEEN

Page SIXTEEN Typeset-10 Installation Guide

Here's a macro to simulate 1/3 - 2/3 proportional leading in a "lead-after" system:

```
[*12lead=[zxa,33][zxb,67][z<->
ma,@][zmb,@][zp@,b][zda,100][v1a]=]<->
[p36,39]Proportional leading must be
computed/1<->
[p10,12]in a lead after system./1/1/1/1<->
[p36,39]Proportional leading must be
computed[*12lead]39@12@/1[p10,12]in
a lead after system./1/1/1/1<->
```

Proportional leading must be computed

in a lead after system.

Proportional leading must be computed

in a lead after system.

APPENDIX A

ERROR MESSAGES

JUSTIF issues messages:

1. On the job TTY (the OPSER terminal at which JUSTIF is run)
2. On the J&H log device (which may be the same as the job TTY.
3. In the .JH file. (These messages may also appear on the log device if so specified in TSYSCF.MAC.)

JOB TTY MESSAGES

MESSAGE	CAUSE	CORRECTIVE ACTION
?DEVICE ASSIGNED TO TTJ CANNOT OUTPUT	J&H log device (TTJ) has been assigned to an input device (e.g., PTR).	JUSTIF has exited. 1. Deassign TTJ or assign TTJ to an output device. 2. Restart JUSTIF.
?SYSTEM ERROR - BUFFER OVERFLOW	Either the line buffer or the hyphenation buffer has overflowed. The buffer is identified in a message on the log device.	
	1. A macro definition or series of definitions exceeds the buffer size. (Line buffer.)	1. Check whether a "=]" sequence was omitted from the end of a macro definition. 2. Separate macro definition with /1 commands. 3. For long macros, remove the definition from the story and create the definition by hand, as follows: a. Using TECO, make a file whose filename is the macro name (characters between * and = in the definition) and whose extension is .MCR. b. Insert the macro definition, replacing the characters up to and including the first = by \, and replacing the last = by \.
	2. Internal JUSTIF error if TJ.JBL in TSYSCF.MAC is too small (line buffer).	1. Change TJ.JBL. 2. Reassemble and restart JUSTIF.

MESSAGE	CAUSE	CORRECTIVE ACTION
	3. Attempt to hyphenate a word longer than 100 characters. (Hyphenation buffer).	1. Increase TH.HBL in TSYSCF.MAC. 2. Reassemble and restart JUSTIF.
ALLOTTING TABLE FOULED	The file ALLOT.TAB has an internal inconsistency.	1. Stop JUSTIF. 2. Check allotting tables (see ALLOT manual). 3. If necessary, restore ALLOT.TAB from a FAILSAFE. 4. Restart JUSTIF.
FILE NOT FOUND J&H CANNOT OPEN INPUT FILE J&H	Story queued to JUSTIF does not exist.	Check story name and queue again.
ENTER FAILURE: STORY ABORTED	JUSTIF cannot create .JH or .BIN files because of protection failure. (Should not occur if JUSTIF runs under [1,2]). Either an existing file of the same name or the entire UFD has too restrictive a protection code.	1. Change UFD or file protection. 2. Requeue story.
?IOERR- SYSTEM ERROR. PLEASE NOTIFY SYSTEMS PROGRAMMER -or- ?FILE ERROR - SYSTEM ERROR. PLEASE NOTIFY SYSTEMS PROGRAMMER.	Fatal file or I/O error. If the "SYSTEM.." portion of the message is not included, JUSTIF continues processing the story. If included, JUSTIF aborts the story and restarts itself.	Save printout from the terminal. Submit a copy of the input text that caused the error together with a copy of the printout and a completed SPR.

LOG DEVICE MESSAGES

MESSAGE	CAUSE	CORRECTIVE ACTION
?LINBUF TOO SMALL FOR LINE	Line buffer overflow	See associated message on job TTY.
?HYPHENATION BUFFER TOO SMALL FOR WORD	Hyphenation buffer overflow.	See associated message on job TTY.
STORY ABORTED	Cause given in previous error message.	See associated message and corrective action.

MESSAGE	CAUSE	CORRECTIVE ACTION
CAN'T ACCESS .RPT FILE	At the end of processing a story, JUSTIF cannot read or write the .RPT file. The story is justified and output but probably will not be allotted.	Try queueing again. Notify systems programmer if error persists.
FILE BEING ALLOTTED. NO NEW .RPT FILE CREATED	A story with the same name and priority is currently being allotted. This story will not be allotted.	Ignore or requeue.
CANNOT FIND OUTPUT MODULE; USING MDJNH	The output driver for the current setup does not exist. The driver MDJNH creates a dummy .JH file for proofing.	Check [d command and allotting tables for the setup. Requeue story.
CANNOT FIND MDJNH.XPN STORY ABORTED	MDJNH does not exist.	Check [d command and allotting tables. Make sure MDJNH.XPN is on DSK: [2001,3].
OUTPUT MODULE TOO LARGE. STORY ABORTED	The output driver is larger than 4K.	If output driver is a DEC module, be sure it has been assembled and loaded correctly. If so, notify systems programmer.
FATAL ERROR READING OUTPUT MODULE	An I/O error occurred when JUSTIF attempted to load the output driver. Story will be aborted.	Requeue story. If error persists, save printout from the terminal. Submit a copy of the input text that caused the error together with a copy of the printout and a completed SPR to a Support Specialist as soon as possible.
QUEUEING FROM FILE filename.extension	Not an error. Normal message when JUSTIF begins processing a story.	None.
filnam.ext/mdate/mtime/cslug	Normal message. Slug line printed on log device for each story if so specified by TJ.SLP in TSYSCF.MAC.	None.
ERRORS:number	Normal message. Reports the number of errors found after processing the story named in the last "QUEUEING" Message.	If any errors, consult .JH file. Correct story and requeue.

MESSAGE	CAUSE	CORRECTIVE ACTION
DONE	Normal message. JUSTIF is finished processing the story named in the last "QUEUEING" message.	None.
ERROR IN DICTIONARY ROUTINE	Serious internal JUSTIF error.	1. Stop JUSTIF. 2. Reassemble and restart JUSTIF. 3. If error recurs, save printout from terminal. Submit copies of printout and the file DSK:WORD.DIC[2001,3] together with a completed SPR form to computer operations manager.
DICTIONARY NOT FOUND IN J&H	File DSK:WORD.DIC[2001,3] does not exist.	1. Stop JUSTIF. 2. Make a file WORD.DIC using TECO. 3. Restart JUSTIF.
ERROR IN DICTIONARY DISK FILE	An entry in the exception word dictionary is not in the proper format (see Appendix B).	1. Stop JUSTIF. 2. Using TECO, correct the word in WORD.DIC. 3. Restart JUSTIF.
%DICTIONARY-CAN'T FIT WORD: word	The exception word dictionary size has been exceeded. The word printed is the first word that would not fit.	1. Stop JUSTIF. 2. Either a. delete excess words from WORD.DIC or b. increase dictionary size in TSYSCF.MAC and reassemble JUSTIF. 3. Restart JUSTIF.

.JH FILE MESSAGES

Error messages in .JH files may emanate from JUSTIF and the output driver for the particular setup.

Messages from JUSTIF

MESSAGE	CAUSE	CORRECTIVE ACTION
MACRO: CALL ASSUMED, 6 CHAR NAME	Macro names may be up to six characters long. After the command sequence [*, more than six characters appear before = or]. The command is processed as a call to the macro whose name is the first characters. Other errors usually result.	Correct the macro command in the story.

MESSAGE	CAUSE	CORRECTIVE ACTION
MACRO: CLOSED ACCESS	System error. JUSTIF cannot read .MCR file because of OPEN failure..	Notify systems programmer.
MACRO: NOT FOUND	A non-existent macro is called. (That is, there is no file macro name .MCR on either the area from which the story was queued or [2001,3].	Check the story to see whether correct macro name is used. Add macro definition to the story if necessary.
MACRO: MAY NOT CREATE	JUSTIF cannot process macro definitions because of protection failure; either a macro of the same name already exists with too restrictive protection or the UFD is protected against creates.	Change UFD or existing macro protection or change macro name in definition.
MACRO: TOO MANY LEVELS	Macros have been nested to more than seven levels.	Change macro calls to decrease nesting.
MACRO: EOF BEFORE END OF DEFINITION	Story ended before the final "=]" of macro definition.	Check macro definition.
ILLEGAL COMMAND	Character "/" or "[" is followed by incorrect letter sequence for composition command. Command ignored. Following characters taken as text.	Correct command in story.
POINTSIZE NOT AVAILABLE	[p or [wp command requests a point size not specified in the setup. Command ignored. Current point size used.	Correct command or ignore.
FACE NOT AVAILABLE	Command requests a face not specified in the setup. May be [f, [fb, [fi, [fs, [wf, [wb, [wi, or [ws.	Correct command or ignore.
LINE LENGTH TOO LONG FOR COMPOSITOR	Measure specified is greater than maximum column for current device. Maximum column width is used.	Correct [c command or ignore
LINE LENGTH TOO SMALL. DEFAULT USED	Measure specified is smaller than the minimum column. Default column width used.	Correct [c command or ignore.

MESSAGE	CAUSE	CORRECTIVE ACTION
NON-EXISTENT DEVICE	Device requested in [o command does not exist.	Correct command or ignore.
INDENTS EXCEED COLUMN WIDTH	Total indent distance (including column, text, hangs, length fits, skews, and paragraph) is greater than the specified measure.	Change indent commands or [c command.
UNDEFINED INDENT	Fixed indent or fixed hang requested which either is not defined in TSYSCF.MAC or uses an indent number other than 0-9.	Change [if or [hf command; or change TSYSCF and reassemble JUSTIF.
ALL ESSENTIAL PARAMETERS NOT DECLARED	A warning. At least one of the five essential parameters is not specified at the beginning of the take. Default(s) used.	Either ignore or add missing [d, [f, [p, [v, or [c command.
SETUP MUST BE FIRST. DEFAULT USED	Take begins with something other than [d command. Default setup used.	Insert [d command or ignore
SETUP ALREADY SPECIFIED	The setup has already been specified by an initial [d command or by default, but a subsequent [d command requests a different setup. Only one setup is permitted for a take. (There should be at most one [d command and it must appear first in the take.) If a second [d command occurs for the same setup, it is ignored and this message is not issued.	Delete or reposition the incorrect [d command.
SETUP DOES NOT EXIST	Setup requested by [d command does not exist in allotting tables. Default setup used.	Either ignore or 1. Stop JUSTIF 2. Check allotting tables. 3. Restart JUSTIF 4. Requeue story.
TOO MANY TAB STOPS	More than 41 tab stops (40 subcolumns) requested. The incorrect tab stop command is terminated, but may have set some tab stops.	Correct tab stop command.
NO SUCH CODE IN XD. DEFINITION NOT COMPLETE	In an [xd command, a character which does not exist has been included in an alternate leader definition. The leader definition is terminated before the illegal character.	Correct [xd command or font table.

MESSAGE	CAUSE	CORRECTION ACTION
6 CHARACTER LIMIT FOR ALTERNATE LEADERS	[xd command uses more than six characters for alternate leaders definition. Remaining characters are interpreted as text.	Correct [xd command.
NULL ARGUMENT ILLEGAL - COMMAND TERMINATED	A zero value is used in a command which does not allow zeros; e.g., [ts. The command is terminated, but may already have performed some action.	Correct command.
ERROR IN OUTPUT ON PREVIOUS LINE	Output driver has detected an error in the output line before this message in the .JH file.	Inspect previous line and correct mark-up error.
LINE OVERSET	JUSTIF could not avoid oversetting this line. Should never occur for hot metal.	Check commands in the story which could overset this line: point size, column measure, indents, tabs, leaders, one-line justify.
FONT FILE NOT FOUND	Font listed in the allotting tables does not exist. All requests for the font will cause FACE NOT AVAILABLE message.	Check allotting tables and font tables on [2001,3].
DEVICE NAME NOT FOUND	JUSTIF cannot find logical device name for this setup in allotting tables. "CANNOT FIND OUTPUT MODULE" message will follow.	Check allotting tables.
DEVICE DOES NOT EXIST	Device number for this setup does not exist in allotting tables. Other messages will follow.	Check allotting tables.
TOO MANY FACES IN GRID TABLE	The number of faces in a grid table within the allotting tables exceeds the maximum specified in TSYSCF.MAC.	1. Change TSYSCF.MAC. 2. Reassemble JUSTIF.
ERROR IN ARITHMETIC COMMAND	Arithmetic command has syntax error.	Correct command.
NUMBER TOO BIG TO SET IN WORDS	[zw command uses value greater than one trillion.	Correct command.

Messages from LINOTRON 505 Driver

MESSAGE	CAUSE	CORRECTIVE ACTION
ILLEGAL POINTSIZE DETECTED AT MD LEVEL	Requested point size is specified in allotting tables, but illegal for 505.	Change point size command or change allotting tables.
ILLEGAL SET WIDTH DETECTED AT MD LEVEL	Requested set width does not match a legal 505 point size.	Change set width command or change allotting tables.
NON-MAGNIFYABLE SET WIDTH REQUESTED IN MAGNIFICATION MODE	Both point size and set width must be in same mode.	Change set width command to a magnified set width, or change point size command to a non-magnified point size.
MAGNIFIED SET WIDTH REQUESTED IN NON-MAGNIFICATION MODE	Both point size and set width must be in same mode.	Change set width command to a non-magnified set width, or change point size command to a magnified point size.
TABLE SEARCH FAILURE - MD MAY BE CLOBBERED	Serious internal driver error.	Reload 505 driver.

Messages from Hot Metal Drivers

MESSAGE	CAUSE	CORRECTIVE ACTION
?COUNT OUT TABLE IN ERROR -or- ?MAGAZINE CAPACITY TABLE IN ERROR -or- ?SPECIAL CHARACTER TABLE IN ERROR -or- ?SPACE TABLE OVERFLOW	All fatal internal errors.	Save printout from the terminal. Submit a copy of the input text that caused the error together with a copy of the printout and a completed SPR to a Software Specialist.
%NO MATS LEFT IN CHANNEL	Too many requests for a character on one line.	Check the repeated character, point size, and leader commands.
%DOUBLE SPACE BAND PROCEDURE ERROR	Two consecutive space bands.	Check text on this line, especially macro calling sequences.
%SPACE BAND FIRST OR LAST IN LINE PROCEDURE ERROR	Line begins or ends with space band.	Check text on this line, especially macro calling sequences.
%NO SPACE BAND PROTECTION IN LEADERS		

APPENDIX B

USING THE HYPHENATION EXCEPTION WORD DICTIONARY

JUSTIF ordinarily uses an algorithm for hyphenation. Occasionally, the algorithm yields an unacceptable hyphenation for a particular word. Proper hyphenations for such words may be entered in the exception word dictionary to ensure correct hyphenation. JUSTIF always consults the exception word dictionary and applies the specified hyphenation in preference to hyphenating a word by algorithm. (If the [yd] command is in effect, the algorithm is never applied and all words are hyphenated according to the dictionary or not at all.)

The exception word dictionary is DSK:WORD.DIC[2001,3]. The maximum size for the dictionary is specified in the system standards. JUSTIF reads the dictionary into core when the program is started. If the dictionary does not exist or exceeds the system standard maximum, JUSTIF prints an error message on the job TTY (See Appendix A). TECO is used to insert words in the dictionary. Words may be upper- or lower-case. A word is terminated by space or carriage return. Hyphens mark the proper hyphenation point(s) in each word. Hyphens after the first character or before the last character of a word are ignored.

Typeset-10 Composition Commands
Summary Index by Function

Command	Page	Format	Comment
Essential Parameters			
Setup	2-1	[d.*x*]	*x*=setup number, defined via ALLOT
Type Face	2-1	[f.*x*]	*x*=type face number
Point Size	2-2	[p.*x*]	*x*=points in point size notation (Sec. 1.7)
Stored Leading	2-2	[v.*x*]	*x*=points in leading notation (Sec. 1.8)
Column Width	2-4	[c.*x*]	*x*=measure in column width notation (Sec. 1.9)
Type Face Change			
Change Face	3-1	[f.*x*]	same as essential parameter
Change Face One-Line	3-4	[wf.*x*]	
Bold Face	3-1	[fb] or /e	uses bold face of current face number
Bold Face One-Line	3-5	[wb]	
Italic Face	3-2	[fi]	uses italic face of current face number
Italic One-Line	3-5	[wi]	
Small Caps Face	3-3	[fs]	uses small caps face of current face number
Small Caps One-Line	3-6	[ws]	
Face Modify (slant)	3-4	[fm]	slants current face
Cancel Slant	3-4	[fm%]	
Face Restore	3-4	[fr] or /d	cancels [fb, [fi, or [fs
All Caps One-Line	3-6	[wc]	
Point Size and Set Size Change			
Change Point Size	4-1	[p.*x*]	same as essential parameter
Change Point Size One-Line	4-1	[wp.*x*]	
Change Set Size	4-2	[es.*x*]	*x*=width in set size notation (Sec. 1.7)
Change Set Size One-Line	4-3	[we.*x*]	
Restore Set Size	4-3	[es%]	cancels [es.*x*]
Leading Change			
Change Leading	5-1	[v.*x*]	same as essential parameter
Change Leading One-Line	5-1	[wv.*x*]	
Cancel Leading One-Line	5-4	[vz]	
Added Lead	5-3	[va.*x*]	*x*=points in leading notation (Sec. 1.8)
Reverse Lead	5-3	[vr.*x*] or [va-*x*]	
Change Vertical Division	5-2	[vd.*x*]	Changes system standard for proportional leading. *x* is percentage of maximum leading on first line.
Column Width Change			
Change Column Width	6-1	[c.*x*]	same as essential parameter
Column Restore	6-1	[cz]	cancels all indents and tabs

Command	Page	Format	Comment
Quads			
Quad Left	7-1	/l	
Quad Right	7-1	/r	
Quad Center	7-1	/c	
Quad Middle	7-2	/q	like /l without line break
Conditional Quad	7-2	/?	/l if no quad after preceding text, otherwise ignored.
Paragraph; Flush and Hang			
Change Paragraph Indent	8-1	[ip.*x*]	*x*=indent in standard or alternate (#) indent notation (Sec. 1.6)
Paragraph Indent	8-1	/p	
Restore Paragraph Indent	8-1	[ip%]	restores system standard
Define Flush and Hang Indent	8-2	[ih.*x*]	*x*=the indent measure in indent notation (Sec. 1.6)
Flush and Hang	8-2	/h	
Cancel Flush and Hang	8-3	[ih%]	
Spacing			
EM Space	9-1	/m	
EN Space	9-1	/n	
Thin Space	9-1	/i	
Fixed Space	9-1	/f	
Figure Space	9-2	/g	
One-Unit Space	9-2	/o	
Thut	9-2	/;	
Bolt	9-2	/:	
EM-Dash	9-2	-	
Pi Case			
Enter Pi Case	10-1	/+	succeeding characters in Pi case until exit Pi
Exit Pi Case	10-2	+,[,/,or]	
One Pi Character	10-1	+	
Tabs			
Set Tab Stops	11-1	[ts.*x,y,z,...*]	*x,y,z* from left margin in standard or alternate indent notation (Sec. 1.6)
Set Equal Tab Stops	11-2	[tc.*x*]	sets tabs for *x* equal subcolumns.
Set Proportional Tab Stops	11-2	[tp.*x,y,z,...*]	sets tabs for subcolumns in ratio *x:y:z*, etc.
Set Tab in Text	11-3	/t	tab set at point of command
Tab Left	11-3	/u	
Tab Right	11-4	/w	
Tab Center	11-4	/v	
Tab Around	11-5	/a	text centered on next tab stop
Cancel Tabs	11-6	[t%]	
Column Indents			
Indent Left	12-3	[il.*x*]	*x* is the indent in standard or alternate indent notation (Sec. 1.6)
Indent Right	12-4	[ir.*x*]	
Indent Both	12-4	[ib.*x*]	
Fixed Indent	12-5	[if.*x*]	*x* is a number (0-9) defined in system standards
Cancel Left Indent	12-6	[il%]	
Cancel Right Indent	12-7	[ir%]	

Command	Page	Format	Comment
Cancel Both Indents	12-7	[ib%]	
Cancel All Indents	12-28	[i%]	cancels all indents: column, text, length fits, hangs, and skews
Text Indents			
Define Text Indent	12-7	[it.*x*]	*x* is a number (0-9)
Left Text Indent	12-8	[ij.*x*]	*x* is point in text defined by [it
Right Text Indent	12-9	[ik.*x*]	
Cancel Left Text Indent	12-10	[ij%]	
Cancel Right Text Indent	12-10	[ik%]	
Hanging Indents			
Left Hanging Indent	12-10	[hl.*x*,*y*,*z*]	*x*=the indent expressed in standard or alternate indent notation *y*=number of lines to wait *z*=number of lines to hang (0=indefinite)
Right Hanging Indent	12-12	[hr.*x*,*y*,*z*]	
Hanging Indent Both	12-13	[hb.*x*,*y*,*z*]	
Fixed Hanging Indent	12-14	[hf.*x*]	*x* is a number (0-9) corresponding to a hang defined in the system standards
Cancel Left Hang	12-14	[hl%]	
Cancel Right Hang	12-14	[hr%]	
Cancel Both Hangs	12-15	[hb%]	
Length Fits			
Left Length Fit	12-15	[ll.*x*,*y*]	*x*=the indent expressed in standard or alternate indent notation *y*=vertical length (picas.points)
Right Length Fit	12-16	[lr.*x*,*y*]	
Length Fit Both	12-17	[lb.*x*,*y*]	
Cancel Left Fit	12-19	[ll%]	
Cancel Right Fit	12-19	[lr%]	
Cancel Both Fit	12-20	[lb%]	
Skews			
Forward Left Skew	12-21	[sl.*x*,*y*]	*x*=maximum indent in standard or alternate indent notation *y*=vertical length to point of maximum indent (picas.points)
Forward Right Skew	12-23	[sr*x*,*y*]	
Forward Skew Both	12-25	[sb.*x*,*y*]	
Reverse Left Skew	12-22	[sl-*x*,*y*]	*x*=maximum indent *y*=vertical distance from maximum indent to full out (picas.points)
Reverse Right Skew	12-24	[sr-*x*,*y*]	
Reverse Skew Both	12-27	[sb-*x*,*y*]	
Cancel Left Skew	12-27	[sl%]	
Cancel Right Skew	12-28	[sr%]	
Cancel Skew Both	12-28	[sb%]	
Mark Point			
Set Vertical Mark	13-1	[vm.*x*]	*x*=mark point number (0-9)
Cancel Vertical Mark	13-1	[vm.*x*%]	cancels definition of mark point *x*
Cancel All Marks	13-2	[vm%]	
Vertical Position	13-2	[vp.*x*]	reverse or forward lead to mark point *x*

Command	Page	Format	Comment
Vertical Columns	13-2	[vc*x*,*y*]	*x*=vertical column width (points) *y*=mark point number Sets columns across paper by auto return to mark point *y* after *x* points of leading. Continues setting columns down the paper until cancelled.
Cancel Vertical Columns	13-4	[vc%]	
Vertical Both	13-4	[vb]	While [vc is in effect, causes immediate column return without waiting for leading.
Vertical Gutter	13-5	[vg*x*]	changes gutter width between columns from system standard default to *x*, expressed in indent notation.
Fractions			
1/8	14-1	/1	
1/4	14-1	/2	
3/8	14-1	/3	
1/2	14-2	/4	
5/8	14-2	/5	
3/4	14-2	/6	
7/8	14-2	/7	
1/3	14-3	/8	
2/3	14-3	/9	
Ragged Mode			
Ragged Left	15-1	[rl]	lines are flush right
Ragged Right	15-2	[rr]	lines are flush left
Ragged Center	15-2	[rc]	lines are centered in column
Ragged Middle	15-3	[rm*x*]	lines are centered *x* points alternately left and right of column midpoint
		[rm]	uses system standard default for ragged middle offset
Cancel Ragged	15-4	[r%]	
Output			
Enable Auto Rule Drop	16-1	[or]	
Cancel Auto Rule Drop	16-1	[or%]	
Cancel Flash	16-2	[of%]	
Restore Flash	16-2	[of]	
Cancel Flash and Lead	16-2	[ob]	one line
Select Output Device	16-3	[o*x*]	device number *x*; generally not used; function of ALLOT
Set Delay	16-3	[op*x*]	*x*=number of tape feeds instead of system standard
Reset Delay	16-4	[op%]	restores system standard
Output Delay	16-4	[od]	outputs tape feeds
Leaders			
Use Leaders	17-1	[xl]	instead of white space
Define Other Leaders	17-1	[xd/]	/=six-character alternate leaders
Use Other Leaders	17-2	[xx]	
Cancel Leaders	17-2	[x%]	
One-Shot Leaders	17-3	/.	
One-Shot Other Leaders	17-3	/x	

Command	Page	Format	Comment.
One-Shot Rule	17-4	/-	
One-Shot White Space	17-4	/s	used in a leader mode
Justification			
Justification Parameters	18-1	[jp*w,x,y,z*]	changes values (relative units) from system standards: *w*=minimum word space *x*=maximum word space before hyphenation *y*=maximum word space before letter spacing *z*=maximum letter space
Allow Letter Spacing	18-2	[jl*x*]	*x*=minimum letter space (relative units)
Cancel Letter Spacing	18-3	[jl%]	
One-Line Justify	18-4	/j	
Hyphenation			
Hyphenation Parameters	19-1	[yp*w,x,y,z*]	changes values from system standards: *w*=minimum number of letters before hyphen *x*=minimum number of letters after hyphen *y*=maximum successive hyphens *z*=minimum number of letters in hyphenated word
Disable Hyphenation	19-2	[ya%]	
Enable Hyphenation	19-4	[ya]	
One-Word Disable	19-4	[yw]	precedes word
Use Dictionary Only	19-6	[yd]	
Discretionary Hyphen	19-6	/-	
Messages			
Tag Line Message	20-1	[mt*x*]	The message *x* is set on film after next quad. Command must have].
Punch Message	20-1	[mp*x*]	Message is punched; must have].
Both Messages	20-2	[mb*x*]	
Macros			
Macro Kill	21-4	[ak]	terminates macro call before all arguments are merged
	21-7	[ak*n*]	terminates *n* levels
Macro Repeat	21-5	[ar*n*]	repeats next macro call *n* times
Arithmetic			
Load Register	22-1	[zx*r,s*]	*r* is a register name *s* is an integer or register *s* is loaded into *r*
Display Register (Arabic)	22-2	[za*s*]	value of *s* output as number
Display Register (Words)	22-2	[zw*s*]	value of *s* output in words
Zero Register	22-2	[zz*r*]	register loaded with zero
Increment Register	22-2	[zi*r*]	One added to register *r*
Display and Increment Register	22-3	[zj*r*]	equivalent to [za*r*][zi*r*]
Add to Register	22-3	[zp*r,s*]	*r*+*s*=*r*
Multiply Register	22-3	[zm*r,s*]	*r*×*s*=*r*
Divide Register	22-4	[zd*r,s*]	*r*/*s*=*r*
Remainder	22-4	[zc*r,s*]	remainder of *r*/*s* is stored in *r*